(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
9 August 2001 (09.08.2001)

PCT

(10) International Publication Number
WO 01/57786 A1

(51) International Patent Classification[7]: G06K 9/00, G06F 17/30

(21) International Application Number: PCT/US01/03557

(22) International Filing Date: 1 February 2001 (01.02.2001)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
09/494,941 1 February 2000 (01.02.2000) US

(71) Applicant: **SCANSOFT, INC.** [US/US]; 9 Centennial Drive, Peabody, MA 01960 (US).

(72) Inventors: **CHEN, Su**; 1284 Montmorency Drive, San Jose, CA 95118 (US). **DONG, Hong**; 10201 Stern Avenue, Cupertino, CA 95014 (US). **KHOUBYARI, Siamak**; 5496 Blossom Vista Avenue, San Jose, CA 95124 (US).

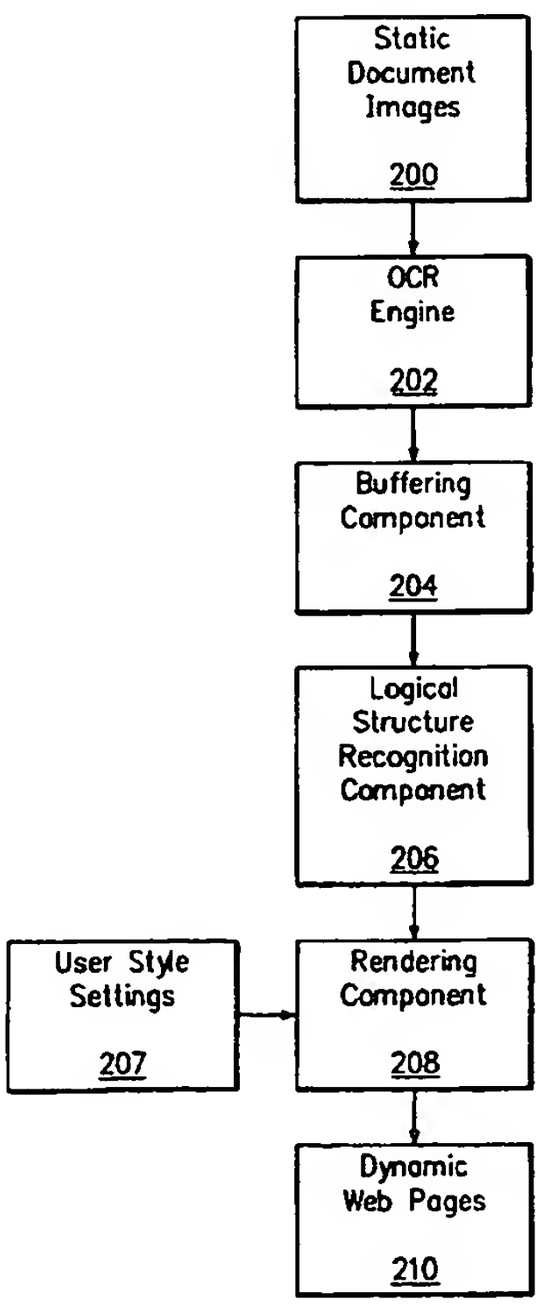
(74) Agent: **PETERSON, James W.**; Burns, Doane, Swecker & Mathis, LLP, P.O. Box 1404, Alexandria, VA 22313-1404 (US).

(84) Designated States *(regional)*: European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE, TR).

Published:
— *with international search report*
— *before the expiration of the time limit for amending the claims and to be republished in the event of receipt of amendments*

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

(54) Title: AUTOMATIC CONVERSION OF STATIC DOCUMENTS INTO DYNAMIC DOCUMENTS



(57) Abstract: The present invention provides a system and method for the automatic conversion of static documents into dynamic documents. In one embodiment, this conversion utilizes images of a static paper document, which may be either scanned into a computer, or loaded into a computer from electronic memory as image files, and converted into dynamic documents, such as documents authored using a markup language. One prominent example of dynamic documents which the present invention is used to create is HTML documents suitable for Internet publication on the World Wide Web (WWW). The present invention utilizes an optical character recognition engine and a logical structure recognition engine to recognize both textual components and structural components of the static paper document, and utilizes a dynamic document rendering component to create dynamic documents, such as HTML documents.

WO 01/57786 A1

AUTOMATIC CONVERSION OF STATIC DOCUMENTS INTO DYNAMIC DOCUMENTS

FIELD OF THE INVENTION

The present invention relates generally to document image processing. More specifically, the present invention relates to pattern recognition, logical structure recognition, image understanding, and multi-modal document presentation.

BACKGROUND OF THE INVENTION

Traditionally, businesses have used paper documents to transfer information, store files, and communicate with clients. As technology has increased, many documents originate on a computer, and may be transferred to paper by way of a printer, or may be transferred electronically by way of a computer network. With increased computing power, integration of paper documents with electronic documents has become increasingly common. Currently, documents may be electronically transferred from computer to computer using a network protocol, such as e-mail. Another alternative is to transfer files from one computer to another by printing a document using a first computer and scanning that printed document using a scanning device connected to a second computer. In this manner, the second computer electronically stores an image of the scanned document.

In order to aid the translation of paper documents into electronic documents, conversion programs such as Optical Character Recognition (OCR) programs have been introduced to recognize typeface characters within a scanned document image. This procedure is known as textual processing, which deals with the text components of a document image. Textual processing typically determines the skew, or tilt at which the document may have been scanned into the computer

and finds columns, paragraphs, text lines, and words. Graphics processing, on the other hand, deals with graphics of a document image, or non-textual components determining line size, thickness, and direction, and the existence of corners and curves.

The contents of a document which has been scanned to put it in electronic form are generally static in nature. More particularly, when the document is first scanned, the result is a monolithic image. After textual and/or graphic processing, the contents of the document are divided into individual elements, e.g. words, lines, etc., that can be edited by the user. However, these elements are independent of one another, and hence the document retains its static quality.

Recently, a new, more dynamic format for electronic documents has become available through the use of markup languages. The document is considered to be "dynamic" in the sense that various components of the documents interact with one another through "metadata," which is essentially information that describes the components of the data and their relationships. For instance, a user can activate an embedded link at one portion of the document, such as an entry in a table of contents, to immediately view another portion of the document. Markup language is a type of language in which dynamic documents are created using metadata. The increasing popularity of the Internet has accelerated the desire for documents in such a format, since it permits users to easily navigate within a single document, as well among multiple documents, and is able to provide functionality within documents such as animation, for example.

Computer users worldwide often desire to publish items on the Internet, using the standard Hypertext Markup Language (HTML) to create web documents. However, in this age of increasing automation, many users expect applications to provide Graphical User Interfaces (GUIs) to allow them to accomplish these types of tasks by having only a high-level concept of the formulation of web pages. For example, many HTML editors are available that allow a user to design, create, and

publish a web page in much the same manner that they are familiar with using a word processing program. Some popular HTML editors include Front Page by Microsoft Corporation, Netscape Navigator® Gold by Netscape Inc., HoTMetaL Pro® by SoftQuad, and Hot Dog by Anna Wave. These products focus primarily on allowing a user to create an HTML document without needing to know how to write HTML code. Using these editors, users may incorporate much of the functionality of HTML code without specific knowledge of the code itself. For example, users may create documents with a hierarchical structure, create tables, import graphics, or create hyperlinks by using the proper commands provided within the GUI.

With all of this automation, desire to create a transparent interface to allow a user to create HTML documents meeting the expectations of automated functionality has dramatically increased in recent years. While HTML editors incorporate much functionality, presently available HTML editors do not automatically recognize the structure of a pre-existing static document and create a correspondingly-structured HTML document. To the contrary, presently available HTML editors require the user to either input the structure, or create it. For example, in the creation of tables in common HTML editors, the user must manually create the table. In the case of hierarchical structure, the user must indicate the heading level of the text being typed within the HTML editor, for example, heading level one, heading level two, and so on. Also, while some presently available HTML editors allow a user to import graphics from an external device such as a scanner, most of these applications do not have a technique to recognize textual characters, or logical structure.

Typical OCR applications, while providing a technique for recognizing textual characters, do not provide a method for recognizing document structure. Applications have recently been developed which recognize some structure within documents. For example, one technique uses template-based publishing to allow

optimization between hard copies and electronic copies of the same document. By this technique, similar templates are used for both types of documents, however, differences which allow for optimization in each of the respective media in which they will be published are provided. For example, font information in an electronic document template may include information such as color, whereas document information in a hard copy, printed version, template may not provide color information if the printer used to print the document is not able to print in color.

Also, the size of each of the templates may be optimized for viewing within each of the documents. For example, the template used to print a hard copy of a document will maintain an aspect ratio corresponding to the aspect ratio of the paper size on which it is to be printed. On the other hand, the template used to create an electronic document will be optimized for the aspect ratio and color combination of the screen on which it is to be displayed. Further modifications to the electronic template may make a document HTML compatible. For example, a font style field in a hard copy template may be modified to include hypertext linking information from the text displayed within that field. In this manner, various text field information may be mapped to different text field information for different types of document displays.

However, one drawback to template-based electronic publishing is that it works well only when the source documents follow a consistent structure. This drawback can prove to be significant, since document structure is typically not consistent among multiple documents from various sources. Because of this drawback, template-based electronic publishing requires careful document pre-planning to correctly structure HTML documents. However, this is not practical when one considers scanned documents, as the documents are often not created by the user who is scanning them, and who, consequently, has no control over their content, style attributes, or format.

One way of recognizing structure within a document has been defined in the Office Document Architecture (ODA), which utilizes geometric structure to provide the hierarchy of a document, by obtaining information on the document based on the manner in which it is presented. Characteristics such as font, font size, character width, number of columns, table information, and graphic information may provide clues as to how a document is structured. The ODA also utilizes logical structure, which derives the hierarchy of a document resulting from breaking a document down into the sections intended by the original author. Examples of this include paragraphs, distinctions between normal and emphasized text, grouping of figures with captions, position of information within tables, and address information. Another structure analyzed with the ODA is referencing structure, which provides information for hypertext links. This type of information includes information such as table of contents information, reference to figure numbers within the text of a document, reference to various sections of text within the text, and references to network addresses. In this manner, ODA may provide various linking relationships between different pages of a document or different sections of a page.

However, to date, applications utilizing the ODA require documents which are overly simplified. Strict rules of structure are used in an ODA system, which may not apply to all types of documents. Again, as with template-based publishing, creating electronic documents within an ODA system requires careful document planning which, as discussed above, is not always the option of the user desiring to add structure to a scanned document. Some examples of such restrictive rules include: labeling text which is top-most as a header, labeling text which bottom-most as a footer, and labeling text which is below a figure and closest to the figure as a caption. It can be appreciated that these rules may not apply a large number of documents, the form of many of which is outside of the control of the user who desires to create structure from scanned documents.

Accordingly, it is desirable to create a system and method that will recognize structure within document images of a variety of different documents without requiring a user to enter document structure as in the case of creating HTML documents, and without requiring a user to carefully pre-plan a document's structure so that it may be recognized within a narrow set of rules or corresponding to a specific type of template. It is also desirable to automate a process whereby a dynamic document, such as an HTML document, may be created from a scanned or other static document image by a user without the user having any specific knowledge as to the structure of the document, and allowing for an automated process, which is transparent to the user, to create the dynamic document without any knowledge of the rules for structuring the document.

SUMMARY OF THE INVENTION

In accordance with the present invention, these objectives are achieved by a system and method that allows a user to scan or otherwise load a static document image, partition the document into zones corresponding to various types of information within a document image, perform optical character recognition (OCR) on the document image, outline the document image by performing logical structure recognition on the structure of the document image, and export a converted document in which structural and hierarchical elements have been recognized.

Further features of the invention and the advantages offered thereby are explained in greater detail hereinafter with reference to specific embodiments illustrated in the accompanying drawings.

BRIEF DESCRIPTION OF THE DRAWINGS

Figure 1 is an exemplary computer system in which the system and method of the present invention may be employed.

Figure 2 is a flow chart of a method performed by the present invention, according to one embodiment.

Figure 3 is a flow chart of a method performed by a logical structure recognition component, in accordance with one embodiment of the present invention.

Figure 4 is a flow chart of the operation of a logical component identification engine, in accordance with one embodiment of the present invention.

Figure 5 is a flow chart of the operation of a heading level assignment component, in accordance with one embodiment of the present invention.

Figure 6 is a flow chart of the operation of a cross-reference identification component, in accordance with one embodiment of the present invention.

Figure 7 is an illustration of a first example of the user interface of one embodiment of the present invention.

Figure 8 is an illustration of another example of the user interface of one embodiment of the present invention.

Figure 9 is an illustration of a further example of the user interface of one embodiment of the present invention.

Figure 10 is an illustration of an additional example of the user interface of one embodiment of the present invention.

Figure 11 is an illustration of a first portion of an HTML document, displayed in a web browser, rendered by one embodiment of the present invention.

Figure 12 is an illustration of a second portion of an HTML document, displayed in a web browser, rendered by one embodiment of the present invention.

Figure 13 is an illustration of a third portion of an HTML document, displayed in a web browser, rendered by one embodiment of the present invention.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

To facilitate an understanding of the principles and features of the present invention, it is explained hereinafter with reference to its implementation in an illustrative embodiment. In particular, the invention is described in the context of software that performs optical character recognition (OCR) functions and logical structural recognition (LSR) functions on a document image, which may be loaded from a variety of peripheral devices including a computer memory device or a peripheral scanning device, and subsequently renders a dynamic HTML document suitable for publication on a network or the Internet. It will be appreciated, however, that this is not the only embodiment in which the invention can be implemented. Rather, it can find utility in a variety of computer configurations as will become apparent from an understanding of the principles upon which the invention is based.

An exemplary computer system of the type in which the present invention can be employed is illustrated in block diagram form in Figure 1. The structure of the computer itself does not form part of the present invention. It is briefly described here for subsequent understanding of the manner in which the features of the invention cooperate with the structure of the computer.

Referring to Figure 1, the computer system includes a computer 100 having a variety of peripheral devices 108 connected thereto. The computer 100 includes a central processing unit 112, a main memory which is typically implemented in the form of a random access memory (RAM)118, a static memory that can comprise a read only memory (ROM) 120, and a permanent storage device, such as a magnetic or optical disk 122. The CPU 112 communicates with each of these forms of memory through an internal bus 114. The peripheral devices 108 include

a data entry device, such as a keyboard 124, and a pointing or cursor control device 102, such as a mouse, trackball, or the like. A display device 104, such as CRT monitor or an LCD screen, provides a visual display of the information that is being processed within the computer, such as the contents of a document. A hard copy of this information can be provided through printer 106, or similar device. Hard copies of documents from other sources can be scanned into the computer to form a document image by way of a scanning device 107, such as a scanner or digital camera. For example, a static document (e.g., MS Word file, WordPerfect file, PostScript document, PDF file, etc.) could be printed to a special print driver that will capture an image of the document, which would not introduce the degradation associated with printing and scanning a document on a printer 106 and scanner 107. Each of these external peripheral devices communicates with CPU 112 by means of one or more input/output ports 110 on the computer. The input/output ports 110 also allow the computer 100 to interact with an external network 128, such as a local area network (LAN) or wide area network (WAN), or the Internet 130.

Computer 100 typically includes an operating system, which controls the allocation and usage of the hardware resources such as memory, central processing unit time, disc space, and peripheral devices. In addition to an operating system, computer 100 may also execute a variety of software applications, thereby adding functionality for the computer user. Computer software applications may reside as a component of the operating system, or may be stored in memory or on any type of machine readable medium, such as disk 122. The software application which performs the automatic conversion of static paper documents into dynamic documents is further described herein in connection with one of the preferred embodiments of the present invention.

Referring to Figure 2, the software application of the present invention performs a method which converts static documents into dynamic documents. In

one preferred embodiment, these dynamic documents comprise web pages, or HTML documents. However, it will be appreciated by those skilled in the art that the present invention need not be limited to the conversion of static documents into HTML documents, as a variety of dynamic documents may be rendered by the present invention using an automatic conversion from static documents. For example, the present invention may be utilized with a variety of document markup languages, some of which provide dynamic links to a user. Some examples include, Extensible Markup Language (XML), Biopolymer Markup Language (BIOML), Standard Generalized Markup Language (SGML), Forms Markup Language (FML), Mathematic Markup Language (MathML), Java Bioinformatic Sequence Markup Language (BSML), Dental Charting Markup Language (DCML), Electronic Data Markup Language (EDML), Pattern Markup Language (PML), Chemical Markup Language (CML), Vector Markup Language (VML), Java Speech Markup Language (JSML), Drawing Markup Language (DrawML), Real Estate Listing Markup Language (RELML), Cold Fusion Markup Language (CFML), TeX, and LaTeX.

Figure 2 illustrates one embodiment of the present invention wherein static document images are converted to dynamic web pages, e.g. HTML documents. In Figure 2, static document images 200 are loaded into the computer's memory 118. Each image may comprise one page of a document, for example, in a bitmap form. A document may originally exist on paper and be converted into electronic form by way of a peripheral scanner 107. Alternatively, the document images may already exist in electronic form and be loaded into the computer by way of a network 128, the Internet 130, or a disk 122. Once the static document image has been loaded into the computer's memory 118, an OCR engine 202 performs optical character recognition on the document image, recognizing known textual characters. Additionally, certain elements of scanned documents, such as graphics, tables, and text styles may be recognized and utilized for rendering the

overall OCR converted document. While multiple pages of static document images 200 may be loaded into the computer's memory 118, OCR functionality is generally performed on a page-per-page basis. It is contemplated that OCR functions may be carried out on multiple pages at one time.

Once the static document images 200 have been recognized by the OCR engine 202, they are passed to a buffering component 204, which stores the OCR information as separate pages. After buffering the OCR information into separate pages, a logical structure recognition (LSR) component 206 analyzes the buffered OCR information, recognizing logical components, heading levels, cross reference information, and various other document attributes in preparation for creating a dynamic document, such as a web page or HTML document. After the LSR component 206 has recognized the general structure of the buffered OCR information, a rendering component 208 subsequently renders a dynamic document utilizing the information from the OCR engine 202, the buffering component 204, the LSR component 206, and user style settings 207 to produce logically structured, hierarchical dynamic documents 210.

In an alternate embodiment, the input document could be a text file, or a combination of textual data and graphical data, rather than a bit-mapped image. In this case, the OCR engine 202 could be replaced by a parser component. This parser component could extract textual and graphical elements directly from electronic files of various formats (e.g., MS Word file; WordPerfect file, PostScript file, PDF file, etc.) and feed this information directly to a buffering component or an LSR component.

Figure 3 illustrates the general operation of the LSR component 206. In Figure 3, text and graphical information that has been buffered into pages by the buffering component 204 is analyzed by the LSR component 206 and is interpreted by a logical components identification engine 302. This engine 302 classifies a physical paragraph, or grouping of symbols, into structural elements such as a

title, heading, body text, header, footer, caption, table, or other element. A basic unit of document structure is a physical paragraph, exceptions being graphics and tables which may contain multiple physical paragraphs. If the document undergoes OCR processing, the individual paragraphs can be identified by the OCR engine 202, and passed on to the engine 302. Once the logical components have been identified by the engine 302, a heading level assignment component 304 assigns a heading level to each heading paragraph identified by the logical component's identification engine 302. A cross-reference identification component 306 identifies cross-references and potential dynamic links, such as email addresses and URL addresses. After the identification of logical components by the logical components identification engine 302, assignment of heading levels by the heading level assignment component 304, and cross-reference identification by the cross-reference identification component 306 are carried out. A markup component 308 subsequently translates the original OCR information pages 300 into tagged OCR information pages 310, in preparation for rendering a dynamic document with the rendering component 208. The markup component 308 indicates the location and label of each of the detected logical components. In this manner, the LSR component 206 assigns a unique label to each basic unit of the OCR information pages, and tags the OCR information pages using a markup component 308 to produce tagged OCR information pages 310.

In one embodiment, the present invention may be configured to allow a user to aid the LSR component 206 in recognizing document structure. This may be accomplished, for example, by a user indicating various preferences. Also, a user may prompt the recognition of various structural elements within a page by defining zones within the page which correspond to various elements. In this manner, a user may define various zones, such as rectangular-shaped zones, irregular-shaped zones, text zones, graphics zones, and table zones, and may define various columns or other formatting structural features to be recognized.

Figure 4 illustrates the operation of the logical components identification engine 302. This engine is essentially a maximum a posteriori (MAP) classifier. This engine receives OCR information pages 300 and uses an attribute gathering component 402 to gather statistics about the document attributes, such as the font and paragraph formatting and style distributions. A probability calibration component 404 uses the distributions gathered by the attribute gathering component 402 to calibrate the pre-defined probability distributions shown in equations 1 and 2 below.

$$P_1 = P(A_i|L,D) \quad (1)$$

$$P_2 = P(A_i|D) \quad (2)$$

In Equations 1 and 2 above, P represents a probability, and P_1 and P_2 represent specific probabilities expressed in Equations 1 and 2 respectively. L represents a label of the paragraph under consideration. A_i represents the i^{th} attribute of the paragraph under consideration, where i may range between 1 and some number, N, and D represents the given document. Examples of attributes which might be taken into account include style (e.g., bold, italic, underline), justification, font, character size, etc. In Equation 1, the probability P_1 represents a conditional probability that a given paragraph attribute, A_i, exists given a paragraph label variable, L, and the document D. In Equation 2, probability P_2 represents a conditional probability that a given paragraph attribute A_i, occurs given that it is within the given document D. These probabilities are utilized by

the probability calibration component 404 in initializing the probabilities to begin with a uniform prior probability shown below in Equation 3.

$$P_3 = P(L|D) \tag{3}$$

The probability P_3 shown in Equation 3 above describes the conditional probability that the paragraph label, L, may occur within a document D. This value is then used to initialize the probabilities to be determined by the probability calibration component 404. Each of these probabilities can be determined empirically, e.g. by statistical analysis of a large number of sample documents.

The probability distributions, P_1, P_2, and P_3, , from the calibration component 404 are used in a Bayesian classifier 406. The Bayesian classifier 406 assigns a posterior probability to each of the possible paragraph labels given all of its observed attributes within the document, as shown below in Equation 4.

$$P(L|A_1,A_2,\ldots,A_N,D) = \frac{P(A_1,A_2,\ldots,A_N|L,D)P(L|D)}{P(A_1,A_2,\ldots,A_N|D)} \tag{4}$$

Equation 4, when simplified by assuming independence among paragraph attributes, yields Equation 5, shown below.

$$P(L|A_1,A_2,\ldots,A_N,D) = \prod_{i=1}^{N} \frac{P(A_i|D,L)}{P(A_i|D)} P(L|D) \tag{5}$$

This classification provides an initial estimate of the posterior probabilities of the paragraph labels, without using the contextual paragraph attributes. It has the benefit of reducing the overall complexity of the calculations carried by the contextual knowledge integration component 408, which applies contextual knowledge experts to improve the initial posterior probabilities estimates.

The contextual knowledge experts applied by the contextual knowledge integration component 408 may include expert systems, intelligent agents, neural-networks, or other techniques devised to apply a set of contextual models to a document to decide where a particular element is intended to fit within the overall structure of a document. Some of the contextual experts used by the contextual knowledge integration component 408 may include, but are not limited to: a header expert, a footer expert, a numbered list expert, a bulleted list expert, a consistency expert, a language-dependent expert, and various other experts.

By applying the various contextual experts, the contextual knowledge integration component 408 produces a modified Bayesian probability estimate, or group of estimates 410. The logical component's identification engine 302 may then process this data again in the probability calibration component 404, the Bayesian classifier 406, and the contextual knowledge integration component 408 to thereby provide revised, modified Bayesian probability estimates. Repeating these functions to achieve revised, modified Bayesian probability estimates, as shown by branch 412 of Figure 4, may be carried out a number of times to provide increasing accuracy. In one embodiment of the present invention, however, one iteration is used in order to provide maximum accuracy with minimal computation power. However, one skilled in the art will recognize that multiple iterations may be performed as computing power allows, to provide ever-increasingly accurate modified Bayesian probability estimates.

Once the desired number of iterations utilizing branch 412 of Figure 4 have been carried out to achieve modified Bayesian probability estimates of satisfactory accuracy, a label assignment component 414 selects a paragraph label that maximizes the modified posterior probabilities. Once the label assignment component 414 has produced identified OCR information pages 416, the information is passed by the logical component's identification engine 302, to the heading level assignment component 304, as shown in Figure 3.

Figure 5 illustrates the general operation of the heading level assignment component 304, shown in Figure 3, which assigns a heading level to each heading paragraph. Any desired number of heading levels can be employed. In the case of HTML documents, for example, six levels are employed. In Figure 5, identified OCR information pages 416 from the label assignment component 414 are processed by an initial heading level assignment component 502, which sorts the heading paragraphs according to their importance in the document, based on the paragraph attributes. Once the paragraphs have been sorted, the initial heading level assignment component 502 assigns an initial heading level value. In the embodiment of the present invention wherein the dynamic documents produced 210 are web pages or HTML documents, the initial heading level value assigned by the initial heading level assignment component 502 may be larger than 6. Once the initial heading levels are assigned, the heading level consolidation component 504 attempts to merge the initial heading level groupings by minimizing a cost function. This merging process continues until an optimal number of heading levels has been reached. In the case of HTML documents, an optimal number of heading levels is 6 or less. Once the heading level consolidation component 504 achieves the optimal number of heading levels, it outputs OCR information pages with heading levels, which is the output of the heading levels assignment component 304 of Figure 3.

When the heading level assignment component 304 outputs OCR information pages with heading levels, the cross-reference identification component 306, also shown in Figure 3, detects cross-references within the pages. Figure 6 shows the general operation of the cross-reference identification component 306, which is configured to detect links, such as email addresses and URL addresses, and find cross-references to figures, tables, and section headings. Within the cross-reference identification component 306, a key word generator 602 processes OCR information pages with heading levels 506 to generate a cross-reference keyword candidate list by determining locations of predefined keywords within the document text. These predefined keywords may correspond to a particular language being used to produce a document. For example, the symbol "@" may be used to determine that an email address is present within a document. Upon generation of a cross-reference keyword candidate list by the keyword generator 602, a pair establishment component 604 establishes pairs between sources and destinations, by attempting to match keywords and identify the types of references they represent, for example, email addresses, URL addresses, and the like, and identify the source and destination for each cross-reference. Upon establishing source-destination pairs, the pair establishment component 604 produces cross-referenced OCR information pages 606. These cross-referenced OCR information pages 606 are subsequently analyzed by the markup component 308 of Figure 3, which tags the pages to produce tagged OCR information pages 310.

Once the tagged OCR information pages 310 are produced, the rendering component 208, shown in Figure 2, is then able to produce dynamic documents, such as web pages or HTML documents 210, based on user defined style settings or pre-defined themes. In one embodiment, the rendering component 208 may employ a multi-modal document presentation. In the case of HTML documents, for example, this multi-modal presentation provides the user with the ability to

view an HTML version of the original document in hypertext format, and the original document image. In this manner, the user can navigate between these views by way of hypertext links and image map files, for example, retaining all of the original information contained within the original document. The image map identifies the particular page, or portion of a page, on which a structural element appears, to thereby retrieve the appropriate image file when the user clicks on a structural element in the HTML view. Any information which may have potentially been lost as a result of, OCR functions, LSR functions, or in any other manner, will still be retained within the original image.

Figure 7 is an illustration of a user interface that can be employed in one embodiment of the present invention. A window 700 contains various on-screen buttons for minimizing, resizing and maximizing the window, scroll bars to scroll within the documents displayed therein, tool bars 710, 712 and 714 whereupon various commands may be represented by on-screen buttons in a variety of forms, and a status bar 728 which is used to display the status of the document or documents being viewed. The window 700 includes various sub-windows, or panes partitioned within the larger window 700. Thumbnail images of all of the pages within the document which have been loaded into the computer memory, either by scanning, from disk, or by some other method, are displayed within a left pane 702. The image of the page currently being displayed, in this case page one, which is highlighted in window 702, is displayed within sub-window 704. The overall hierarchical structure of the entire document, encompassing all of the pages displayed within sub-window 702, is shown within sub-window 706. The dynamic document, in this case an HTML document, is shown in sub-window 708.

The standard tool bar 710 allows a user to depress various on-screen buttons to accomplish functions within the application, such as beginning a new document, opening a file, saving the current document, printing the active

window, or a variety of other functions. The zone tool bar 712 allows a user to perform various functions relating to the computer's recognition of zones on a page. Using the zone tool bar 712, the user may, for example, draw zones within a scanned document to influence how the computer interprets the structure of the document. For example, the user may draw rectangular, irregular, row, or column zones, or may add to, subtract from, or reorder zones within the document image. The user may also define particular zone properties allowing for greater customization in their interpretation by the logical structure recognition component 206.

An AutoWeb tool bar 714 contains a variety of different buttons, which allow for convenient access to functions contained within the application. For example, an Auto command button 716 allows a user to select between an AutoWeb configuration or a web wizard configuration, wherein a user may be prompted by the application to input various data in order to achieve a particular configuration within a finished, dynamic document. A load image button 718 allows a user to select between retrieving an image file from a disk, or capturing an image using a peripheral scanning device. A zone command button 720 allows a user to select between a variety of different types of documents which he or she wishes to scan. For example, a user may select between single-column pages, multiple-column pages, spreadsheet pages, and mixed pages. In this manner the user may provide the application with information concerning the general form of the document about to be loaded into the application, thereby providing a way for the computer to more effectively utilize contextual experts to determine the logical components of the document. The OCR command button 722 allows the user to choose between performing OCR functions and performing OCR functions with proofreading capabilities. The export command button 726 allows the user to choose between various saving and exporting options, such as saving a document in various file formats, saving the document in an HTML format and launching a

browser in which to view the document, or deferring export of the document until a later time.

The status bar 728 provides the user with various information such as the page currently being viewed, the type of hierarchical structure currently being highlighted, and various other informational items. Within sub-window 702, thumbnail images of all of the pages of the document are displayed. For example, a thumbnail of the first page 730 is displayed, with a page indication 732 in the lower left-hand corner and a status indication icon 734 in the lower right-hand corner. The status indication icon 734 changes as the page for which it is being displayed undergoes various operations. In Figure 7, the status indication icon 734 is in the form of eyeglasses representing that the document has been recognized using OCR functionality. This icon changes to represent, for example, that the document has been outlined, or separated into zones.

The title of the page, "HotFudge Business Plan", 735 is displayed within sub-window 704, which shows the original scanned image of page one. The second item 736 on the scanned image of page one contains address information, and authorship information. Each of these elements 735, 736 are displayed in various forms within the other sub-windows. For example, the title 735, from the original document image shown in sub-window 704, is also displayed in sub-window 708, which shows the HTML version of the document as the HTML title 738. Also the title 735, from the original document shown in sub-window 704, is displayed in sub-window 706, as the document title 740. This is true also for additional remaining information on this and other pages within the document. For example, address and authorship information 736 shown in sub-window 704 is also displayed in sub-window 706, wherein an email address is recognized and identified by an envelope icon and a URL is recognized as an Internet address and identified by a world icon.

Sub-window 706 illustrates the general structure of the overall document including each of the pages shown in sub-window 702. Various heading levels are shown spanning all pages of the document. For further convenience to the user, an outlying tool bar 742 is provided within sub-window 706. However, this tool bar could also be incorporated within the general tool bars of window 700. This tool bar provides the user with the ability to promote or demote various structural items by increasing or decreasing their priority within the document, or changes various items to headers and footers, delete certain items, and filter objects.

Figure 8 is an illustration of a second example of the user interface of Figure 7. In window 800 a different page of the same document that is displayed within window 700 is shown. Like the window 700 shown in Figure 7, window 800 comprises several sub-windows 802, 804, 806, and 808 within the larger overall window. The overall view of the document, along with thumbnail images of each of the pages within the document, is shown within sub-window 802. The original scanned document image, with outlining indications, is shown within sub-window 804. The overall outline of the dynamic document, in this case an HTML document, is shown within sub-window 806. The rendered dynamic document, in this case a web page or HTML document, is shown within sub-window 808. Each of these sub-windows utilize scroll bars, on-screen buttons, and are adjustable in size to allow for convenience in manipulation by the user.

The view of the document image shown in sub-window 804 corresponds to page 7 of the overall document. This information can be ascertained by viewing sub-window 802, wherein the thumbnail image of page 7 of the document is highlighted. As before, the thumbnail images of the pages of the document indicate page number, and status, by way of a status icon. As can be seen in the thumbnail image 830 within sub-window 802, this page of the document contains text and a small image, which corresponds to the image shown in sub-window 804 near the top of the page.

The structure of the document, whose image is shown in sub-window 804, is outlined in sub-window 806. For example, a header 810 has been identified and indicated within sub-window 806 as the header 812. Near the text of the header 812 within the document outline shown in sub-window 806, is an icon that is made to represent a header within a page. As can be seen in sub-window 808, this header is rendered as a hypertext header 814.

The structure of the document outlined in sub-window 806 is further organized, in that the main heading 816 on the document page, which is shown in sub-window 804, is assigned the highest heading level, H1, as shown by the representation of the title 818 within the document outline contained in sub-window 806, which contains an H1 indicator next to it. On this page of the document, each subheading is assigned a lower priority, or lower heading level, within the same page. The heading is rendered within the HTML document shown in sub-window 808 as the largest text 820 on this page of the HTML document.

Within the text contained immediately below the heading 816 of the section on page 7 of the document shown in sub-window 804 is a reference 822 to Figure 5.1. This reference is recognized by the cross reference identification component 306, shown in Figure 3, which forms part of the LSR component 206. Once the reference has been identified by these components, a link is formed between the referring text 822 and the item to which it refers. The link is represented within the overall document outlined in sub-window 806 as the link 824, which displays the text, "Figure 5.1" next to a graphic representing the link, in this case, in the representation of links of a chain. This link then becomes a hypertext link 826 within the HTML document displayed in sub-window 808, which refers to the image 828 shown in the document displayed within sub-window 804 as Figure 5.1. This Figure 5.1 is also represented as a graphic 832 within the document outline contained in sub-window 806, which provides an indication that a graphic is contained within the page and indicates the page and paragraph number where it is

located, in this case on page 7, in the second paragraph. Next to the reference 832 indicating that the document contains a graphic, an icon representing a graphic is displayed, for ease in identification to a user. This graphic, Figure 5.1, though not shown in sub-window 808, is also rendered within the HTML document, in a format germane to HTML documents, for example JPEG or GIF format. However, other formats could be rendered which correspond to various other markup languages in which the dynamic document shown in sub-window 808 could be rendered.

The subheading 834 "5.1 Domestic Packaging" shown within the document image contained in sub-window 804, is recognized as a subheading below the heading 816. An indication 836 that this subheading is assigned heading level 2, H2, is shown within the document outline contained in sub-window 806. In this manner, the organization of the overall document is preserved. Although not shown in sub-window 808, the subheading 834 indicated by the organizational element 836 is rendered as a subheading within the HTML document shown in sub-window 808.

In Figure 9, a further example of the user interface in accordance with the embodiment of Figure 7 is shown. In this figure, page 8 is the page currently being displayed, which is indicated by the highlighted thumbnail 930 of page 8. In the original document image contained within sub-window 904, the header 910 is indicated within the document outline shown in sub-window 906, as element 912. This header is created within the HTML document as an HTML header 914 displayed within sub-window 908.

The subheading 916 which is a subheading under the heading 816 shown in Figure 8 is assigned heading level 2, H2, as shown in conjunction with element 918 displayed in the document outline of sub-window 906. Also contained within the original document image is a table, 922, which has been recognized by the LSR component 206. This table 922 is indicated within the document outline as

element 924, which assigns the name of the first cell as a title, and displays an icon representing a table in sub-window 906. The table is rendered within the HTML document as table 926, shown in sub-window 908. It will be recognized by those skilled in the art, that the icons displayed within Figures 7-9 are illustrative only, and that a variety of different iconic representations of headings, headers, footers, links, graphics, and tables may be displayed utilizing various different graphical forms. Furthermore, it will be recognized by those skilled in the art that, while certain headings have been assigned within the overall outline of the document shown in sub-windows 706, 806, and 906, Figures 7-9 represent only one manner in which these elements may be named. For example, it is contemplated that a user may intervene and customize the titles and names of the elements shown within the document outline, shown in sub-window 906, for example.

In Figure 10, another example of the user interface in accordance with the embodiment of Figure 7 is shown. In Figure 10, the same page of the overall document shown in Figure 8, namely page 7 of the document, is shown. This figure illustrates that all views of a document need not be shown at all times. In window 1000, the images of each page of the document are not displayed in a sub-window. However, the original, outlined document is displayed within sub-window 1004, the document outline is displayed within sub-window 1006, and the rendered, dynamic document, in this case an HTML document, is displayed within sub-window 1008. It will be appreciated by those skilled in the art that the various sub-windows may be selectively displayed within the window 1000 as desired by the user. The windows which are to be displayed may be selected by the user or by various default commands programmed into the application to provide convenience for a user. A user may wish, for example, not to view the HTML rendered document shown in sub-window 1008. Accordingly, a user may select an option which hides this window, or may resize the window to a decreased size.

In Figure 11, an HTML document rendered by the present invention is shown displayed in a web browser window 1100. In this case, the web browser used is Microsoft's Internet Explorer®; however, it will be recognized by those skilled in the art that a variety of different browsers, such as Netscape Navigator®, which are capable of displaying an HTML document could be used to view the document. The HTML document rendered by the present invention is created in a manner such that the size of the screen is utilized to optimize the amount of information displayed to the user within a single frame. It will be noted that the information displayed in Figure 11 corresponds to pages 1 and 2 of the document shown in Figure 7. Rather than only displaying page 1, which contains minimal information, the present invention optimizes the amount of information shown to create a web page that displays the maximum possible information on one web page of the document. It will be appreciated by those skilled in the art that the manner in which the present invention determines page breaks within the HTML document may be set as part of a group of user preferences, or may form part of a group of pre-determined rules. These web pages are displayed within a sub-window 1102 contained within the web browser 1100.

Also for the convenience of the user, a table of contents is created and displayed within sub-window 1104 of the web browser 1100. This table of contents is created from the document outline, displayed within sub-window 706 of Figure 7, for example. The HTML document is displayed with the title 1106, which corresponds to the title 735 shown within the original image of the document contained in sub-window 704 in Figure 7, within the document outline shown in sub-window 706 as element 740, also in Figure 7, and the title 738 within the HTML document rendered by the present invention, shown in sub-window 708 of Figure 7. The email address, which has been recognized by the present invention, is created as a hypertext link 1108 rendered by the rendering component 208, shown in Figure 2, which provides a user with an easy means for

sending an email message to the address listed. Also recognized by the present invention is a URL address 1109, which has also been rendered as a hypertext link for the convenience of the user. The first heading "1 Introduction" 1110 is displayed beneath the document title.

Within sub-window 1104, the table of contents contains various links to the heading levels within the document. For example, when selected, a link to the document title 1112 allows a user to move and to display this portion of the document within sub-window 1102. Similarly, the link to the introduction 1114, may be selected to display the corresponding portion of the HTML document within sub-window 1102. Also for the convenience of the user, various navigational hypertext links are displayed at the top and bottom of page 1116, and 1118. The two links shown on this page are links to the table of contents, which allows a user to display the table of contents within the main sub-window 1102, and the next page of the HTML document that follows in sequence after the page currently displayed in sub-window 1102. Additional navigational hypertext links, such as a link to the previous page of the HTML document, can also be displayed, as appropriate. However, in the case of Figure 11, since the page of the HTML document being displayed is the first page, there is no need for a link to the previous page. It will be recognized by those skilled in the art that navigational links contained at the top and the bottom of the page 1116, and 1118, need not be limited to links to previous and following pages, and table of contents. Rather, dynamic links to various important pages, graphics, tables, or other important items within the document may be displayed as navigational hypertext links.

Figure 12 is an illustration of a user interface, wherein an HTML document rendered by the present invention is displayed within web browser 1200. The portion of the document being displayed in Figure 12 corresponds to the portion of the document displayed in Figure 8, namely page 7 of the document. The web browser 1200 is divided into sub-windows, including a main sub-window 1202 and

a table of contents sub-window 1204. In the main sub-window 1202, a main heading 1206 is displayed. Also, a reference to Figure 5.1 has been identified by the present invention and a hypertext link 1208 has been created from the text to the image 1210. As shown in the document outline contained within sub-window 806 of Figure 8, the heading 1206 is followed by several subheadings, for example, subheadings 1212 and 1214. The heading and subheadings are also shown within the table of contents displayed within sub-window 1204 as hypertext links 1216, 1218, and 1220, respectively. A user may, by selecting one of these hypertext links within the table of contents sub-window 1204, view the portion of the document corresponding to these headings or subheadings within the main sub-window 1202.

In Figure 13, a view of page 8 of the document, also shown in Figure 9, is displayed within a sub-window 1302 of a web browser window 1300. Also contained within the web browser window 1300 is a table of contents sub-window 1304 which contains the same table of contents shown in sub-window 1104 of Figure 11 and sub-window 1204 of Figure 12. Two subheadings 1306 and 1308 are displayed within sub-window 1302, and corresponding links 1312 and 1314 are contained within the table of contents. Also displayed within sub-window 1302 is a table 1310 which corresponds to the table contained within the original document 922 shown in Figure 9. A user, currently viewing a different page of the document within sub-window 1302, could view the table 1310 by selecting link 1312 contained within the table of contents displayed in sub-window 1304. This type of HTML document rendering is convenient because it requires minimal effort on the part of the user, and provides maximum functionality to potential clients accessing the information on the Internet.

The convenience of the present invention is in its ability to automatically create and render dynamic documents, such as HTML documents, from static documents, which may be loaded from a scanner or an electronic source. A user

may, by depressing four buttons 718, 720, 722, 724, and 726, shown in Figure 7, create from a piece of paper, an HTML document suitable for publishing an Internet document on the World-Wide Web (WWW). This provides a user with maximum convenience and maximum flexibility in creation of such dynamic documents from static documents.

From the foregoing, it can be seen that the present invention provides a system and method for creating dynamic documents automatically from static documents. In one embodiment, this is accomplished as document images may be scanned or loaded into the application's memory, and functionality such as optical character recognition, logical structure recognition and hypertext rendering, may be carried out on the document images to produce an HTML document. However, as previously discussed, the present invention need not limit itself to this one embodiment. The present invention may, for example, be used in conjunction with a variety of different markup languages, web browsers, and document sources. The presently disclosed embodiments are, therefore, considered in all respects to be illustrative and not restrictive. The scope of the invention is indicated by the appended claims, rather than the foregoing description, and all changes that come within the meaning and range of equivalence thereof are intended to be embraced therein.

WHAT IS CLAIMED IS:

1. A method for the automatic conversion of a static document into a dynamic document, comprising the steps of:

loading a static document into computer memory;

analyzing and recognizing the structure of the static document utilizing a logical structure recognition (LSR) component;

rendering code associated with a dynamic document format in accordance with the recognized structure of the static document; and

creating a dynamic document utilizing the rendered code.

2. The method of claim 1, wherein said step of rendering is based on user-defined style settings or pre-defined themes.

3. The method of claim 1, wherein said steps of rendering code and creating a dynamic document are performed in accordance with user preferences.

4. The method of claim 1, wherein said static document comprises an electronic file, further comprising the steps of:

recognizing the textual and graphical components of the static document utilizing a file parser to create information pages from the static electronic file; and

buffering the information pages using a buffering component.

5. The method of claim 1, wherein said static document comprises a document image, which is loaded into computer memory, and further comprising the steps of:

recognizing the textual components of the static document utilizing an optical character recognition (OCR) engine to create OCR information pages from the static document image; and

buffering the OCR information pages using a buffering component.

6. The method of claim 5, wherein the static document image is scanned into the computer utilizing a peripheral scanner.

7. The method of claim 1, wherein said static document is loaded into a computer memory from an electronic storage device.

8. The method of claim 5, wherein said static document is loaded into computer memory from a magnetic storage device.

9. The method of claim 1, wherein the step of rendering code associated with the dynamic document comprises rendering hypertext information to create hypertext markup language (HTML) documents.

10. The method of claim 1, wherein the step of analyzing and recognizing the structure of the static document comprises the steps of:

identifying logical components of the static document;

assigning heading levels;

identifying cross references contained within the static document; and

tagging the static document with markup language components to produce a tagged static document.

11. The method of claim 10, wherein said step of identifying logical components of the static document comprises the steps of:

(a) gathering document attribute distributions;

(b) calibrating Bayesian probability distributions of said document attribute distributions;

(c) assigning a posterior probability to each of the calibrated Bayesian probability distributions assigned;

(d) applying contextual knowledge experts to the posterior probabilities assigned in said step of assigning;

(e) modifying the Bayesian probability distributions according to said contextual knowledge experts;

(f) selecting a label associated with a Bayesian probability distribution determined by a maximum a posteriori (MAP) classification; and

(g) creating an identified static document containing labels determined by the MAP classification.

12. The method of claim 11, wherein steps b-e are repeated until a modified Bayesian probability estimate of a pre-determined accuracy is achieved.

13. The method of claim 12, wherein steps b-e are repeated for one iteration.

14. The method of claim 10, wherein said step of assigning heading levels further comprises the steps of:

analyzing an identified static document;

assigning an initial heading level assignment to various elements contained within the identified static document; and

consolidating the heading level assignments through optimization procedures to create a static document with heading levels.

15. The method of claim 14, wherein said step of consolidating the heading level assignments is configured to create an optimal number of heading levels within a document corresponding to a document standard.

16. The method of claim 15, wherein said document standard is an HTML standard and the maximum number of heading levels to be consolidated is 6.

17. The method of claim 10, wherein said step of identifying cross references further comprises the steps of:

analyzing a static document with heading levels;

generating a cross reference keyword candidate list;

establishing a source and destination pair for each keyword of said keyword candidate list; and

creating a cross-referenced static document.

18. A system for automatically converting a static document into a dynamic document comprising:

a document reader which loads a static document into computer memory;

an LSR component which analyzes and recognizes the structure of the static document pages;

a rendering component which renders code associated with a dynamic document; and

an output mechanism, which creates dynamic documents utilizing the code generated by the rendering component.

19. The system of claim 18, wherein said rendering component renders code based on user-defined style settings or pre-defined themes.

20. The system of claim 18, wherein said document reader is configured to load electronic versions of said static document, further comprising:

a parser which recognizes the textual and graphical components of an electronic version of the static document and creates information pages from the static document; and

a buffering component which buffers the information pages.

21. The system of claim 18, wherein said document reader is a document image reader configured to load images of said static document images, further comprising:

an OCR engine which recognizes the textual components of the static document and creates OCR information pages from the static document images; and

a buffering component which buffers the OCR information pages.

22. The system of claim 21, wherein said document image reader is a peripheral scanner.

23. The system of claim 18, wherein said document reader is an electronic storage device reader.

24. The system of claim 18, wherein said document reader is a magnetic storage device.

25. The system of claim 18, wherein said rendering component renders hypertext information to create hypertext markup language (HTML) documents.

26. The system of claim 18, wherein said LSR component further comprises:

a logical components identification engine;

a heading level assignment component;

a cross reference identification component; and

a markup component which produces a tagged static document.

27. The system of claim 26, wherein said logical components identification engine further comprises:

an attribute gathering component;

a probability calibration component;

a Bayesian classifier;

a contextual knowledge integration component;

a modified Bayesian Probability estimator;

a label assignment component; and

a mechanism for outputting an identified static document.

28. The system of claim 27, further comprising an iterator.

29. The system of claim 26, wherein said heading level assignment component further comprises:

an analyzing component which analyzes the identified static document;

an initial heading level assignment component; and

a heading level assignment consolidation component.

30. The system of claim 26, wherein said cross reference identification component further comprises:

an analyzing component which analyzes the static document with heading levels;

a keyword generator;

a pair establishment component; and

an output component which outputs a cross-referenced static document.

31. A computer program contained on a computer-reliable medium comprising the steps of:

loading a static document into computer memory;

analyzing and recognizing the structure of the static document utilizing a logical structure recognition (LSR) component;

rendering code associated with a dynamic document to be created utilizing a rendering component; and

creating a dynamic document utilizing the code generated by the rendering component.

32. The computer program of claim 31, wherein said step of rendering code is based on user-defined style settings or pre-defined themes.

33. The computer program of claim 31, wherein said steps of rendering code and creating a dynamic document are performed in accordance with user preferences.

34. The computer program of claim 31, wherein said static document comprises an electronic file, further comprising the steps of:

recognizing the textual and graphical components of the static document utilizing a file parser to create information pages from the electronic file; and

buffering the information pages using a buffering component.

35. The computer program of claim 31, wherein said static document comprises a document image which is loaded during said loading step, further comprising the steps of:

recognizing the textual components of the static document utilizing an optical character recognition (OCR) engine to create OCR information pages from the static document image; and

buffering the OCR information pages using a buffering component.

36. A system for automatically converting a static document into a dynamic document comprising:

means for loading a static document into computer memory;

means for analyzing and recognizing the structure of the static document utilizing a logical structure recognition (LSR) component;

means for rendering code associated with a dynamic document to be created utilizing a rendering component; and

means for creating a dynamic document utilizing the code generated by the rendering component.

37. The system of claim 36, wherein said means for rendering renders code based on user-defined style settings or pre-defined themes.

38. The system of claim 36, wherein said means for loading a static document comprises images, further comprising:

means for recognizing the textual components of the static document utilizing an optical character recognition (OCR) engine to create OCR information pages from the static document images; and

means for buffering the OCR information pages using a buffering component.

39. A logical structure recognition (LSR) engine which analyzes and recognizes the structure of static documents, comprising:

a logical components identification engine;

a heading level assignment component;

a cross reference identification component; and

a markup component which produces a tagged static document.

40. The LSR engine of claim 39, wherein said logical components identification engine comprises:

an attribute gathering component;

a probability calibration component;

a Bayesian classifier;

a contextual knowledge integration component;

a modified Bayesian Probability estimator;

a label assignment component; and

a mechanism for outputting an identified static document.

41. The LSR engine of claim 40, further comprising an iterator.

42. The LSR engine of claim 39, wherein said heading level assignment component comprises:

an analyzing component which analyzes the identified static document;

an initial heading level assignment component; and

a heading level assignment consolidation component.

43. The LSR engine of claim 39, wherein said cross reference identification component comprises:

an analyzing component which analyzes the static document with heading levels;

a keyword generator;

a pair establishment component; and

an output component which outputs a cross-referenced static document.

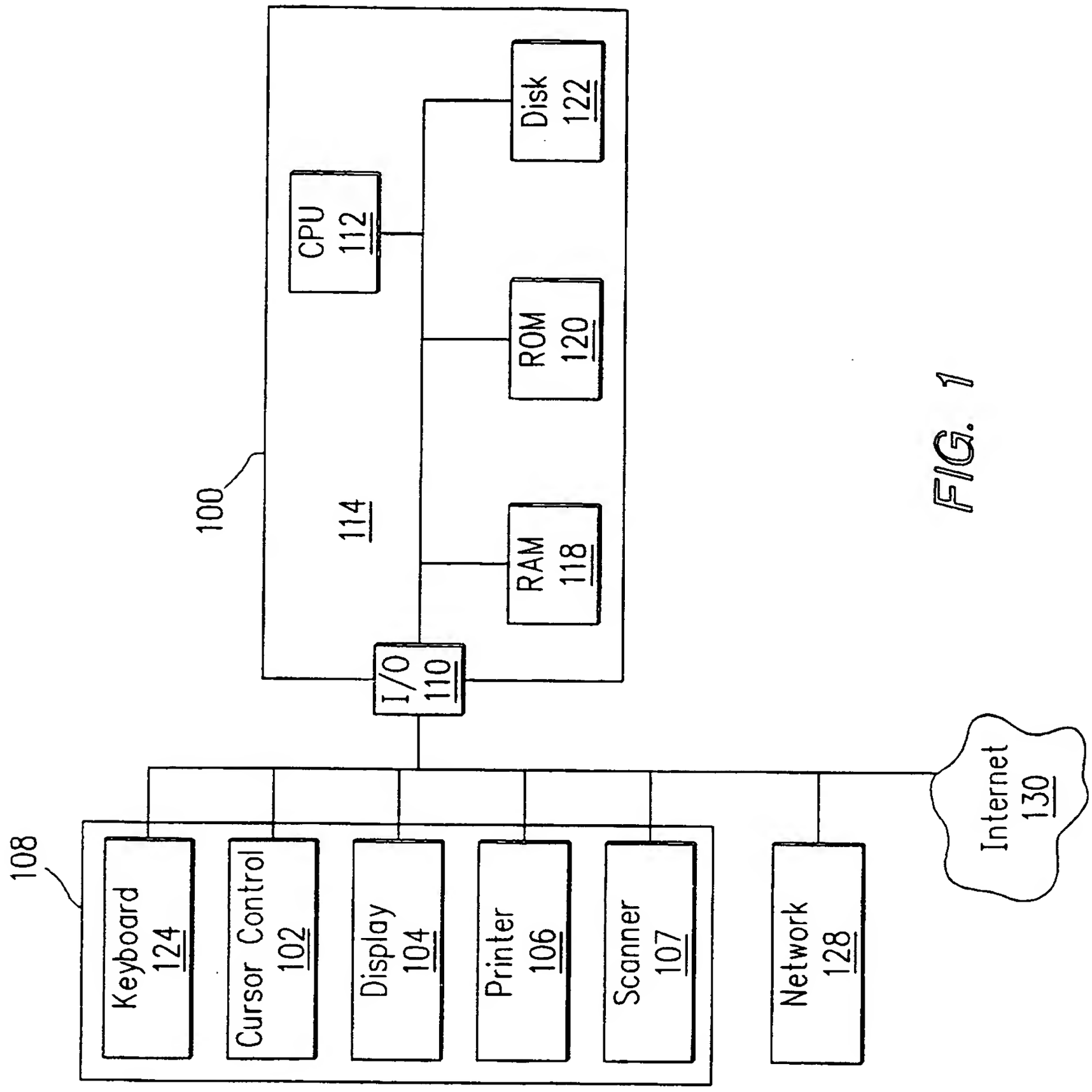

100
114
CPU 112
Disk 122
ROM 120
RAM 118
I/O 110
108
Keyboard 124
Cursor Control 102
Display 104
Printer 106
Scanner 107
Network 128
Internet 130

FIG. 1

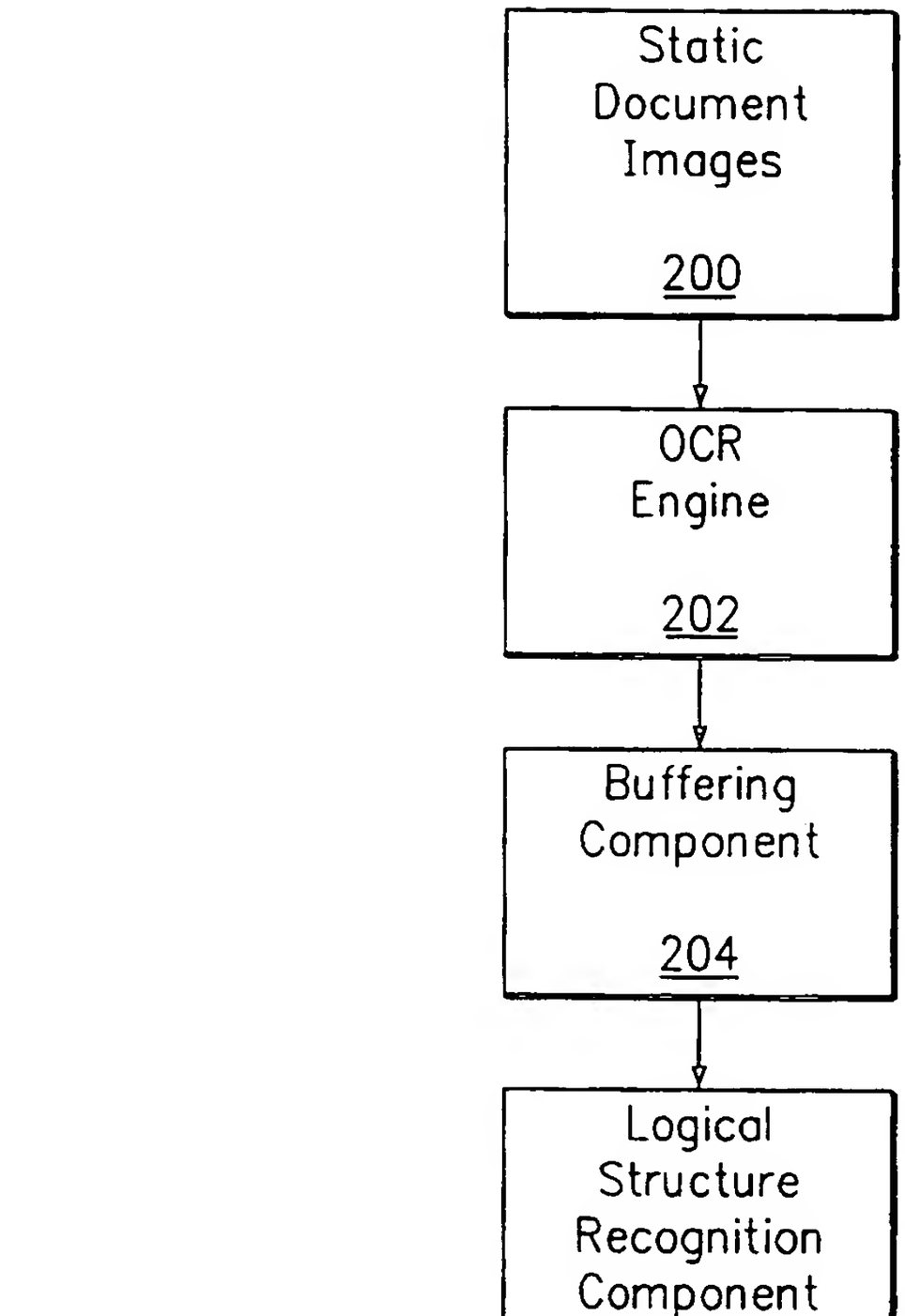

Static Document Images
200
OCR Engine
202
Buffering Component
204
Logical Structure Recognition Component
206
User Style Settings
207
Rendering Component
208
Dynamic Web Pages
210

FIG. 2

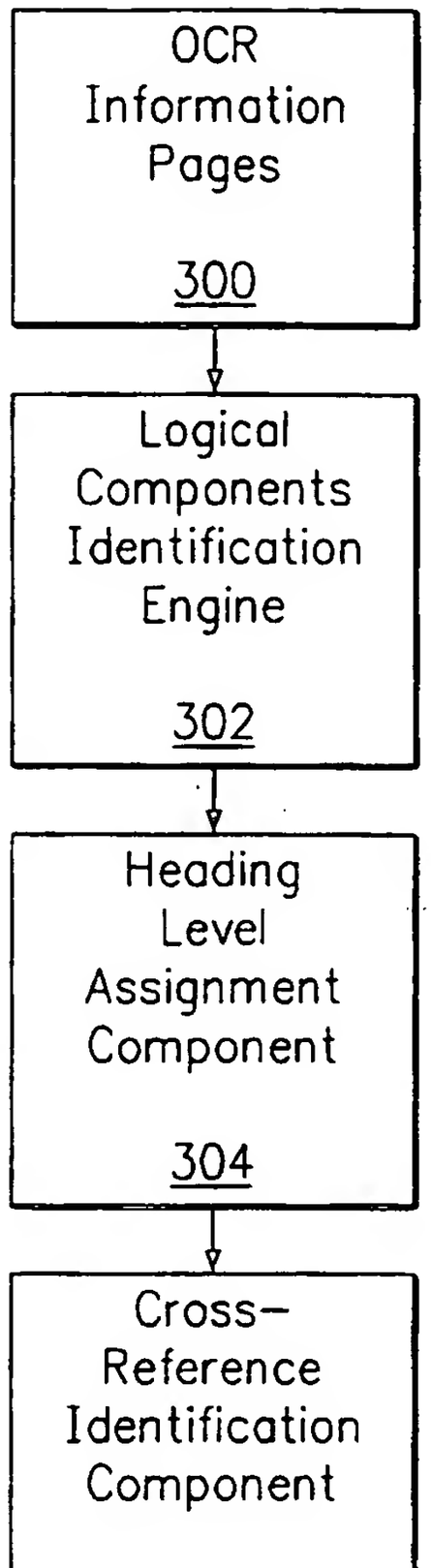

OCR Information Pages
300
Logical Components Identification Engine
302
Heading Level Assignment Component
304
Cross-Reference Identification Component
306
Markup Component
308
Tagged OCR Information Pages
310

FIG. 3

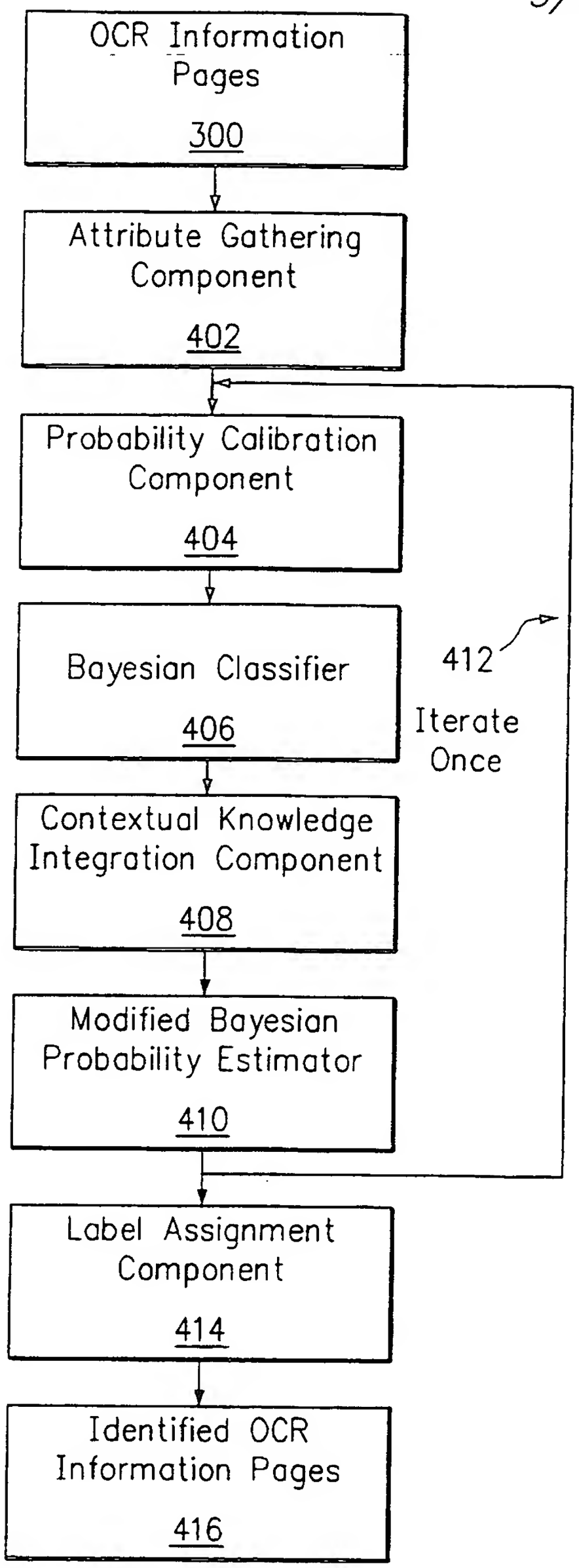


FIG. 4

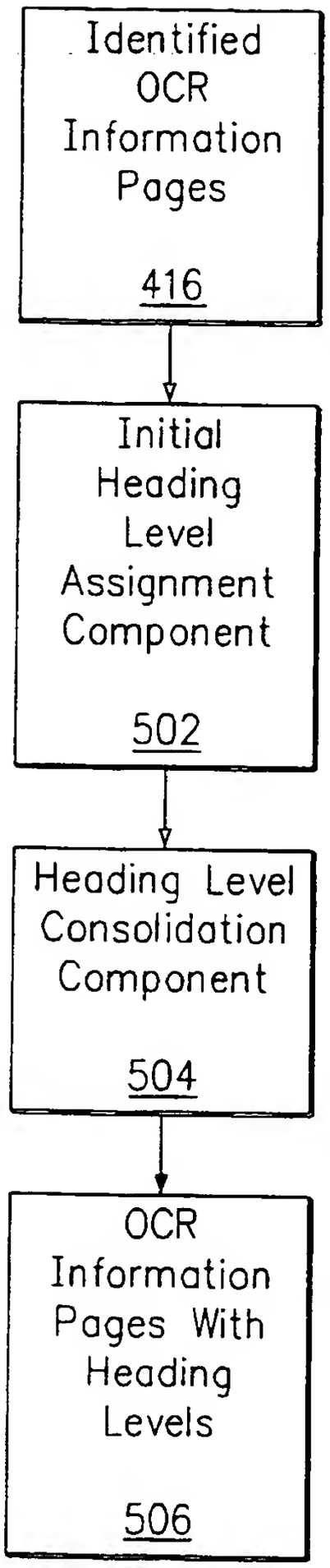


FIG. 5

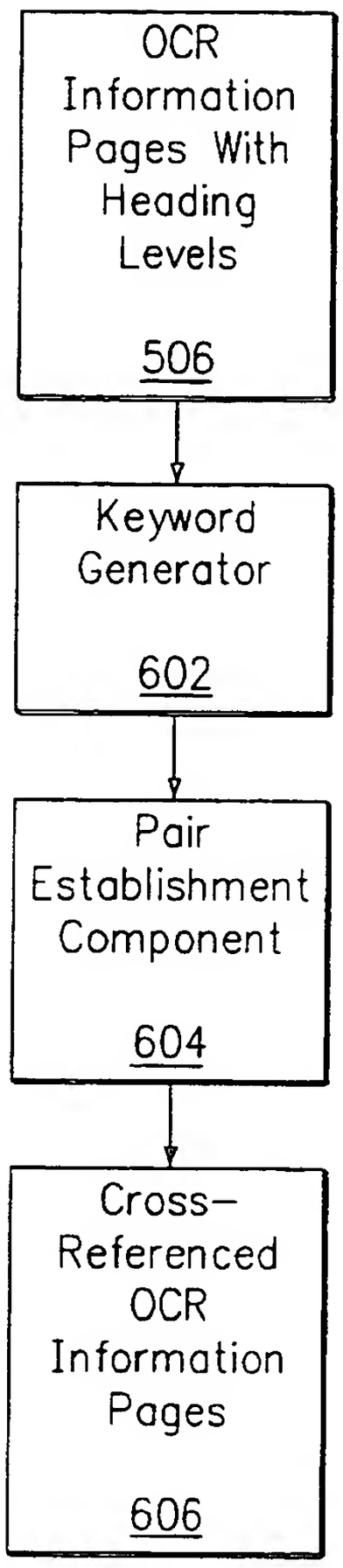


FIG. 6

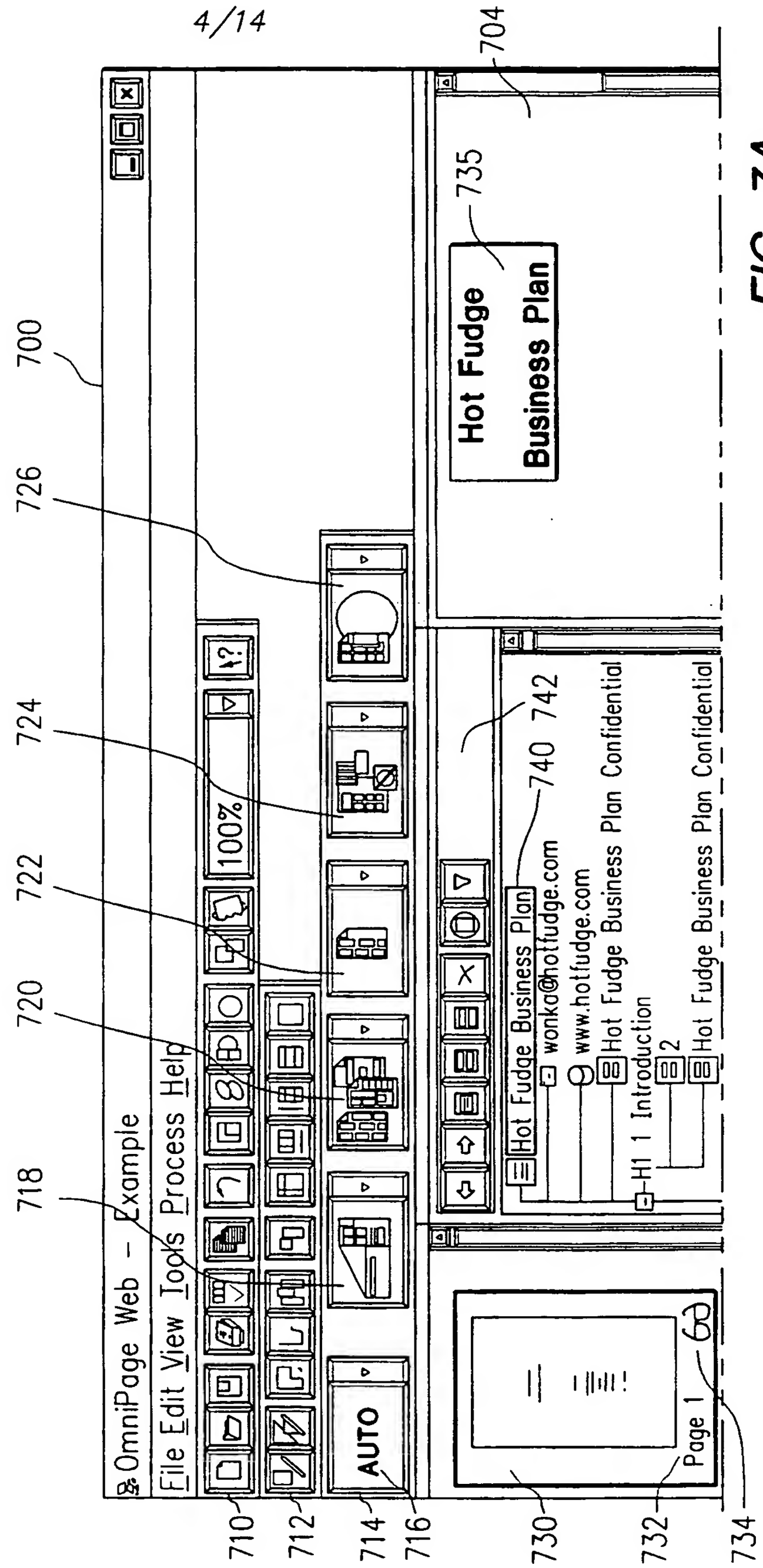


FIG. 7A

FIG. 7B

FIG. 7

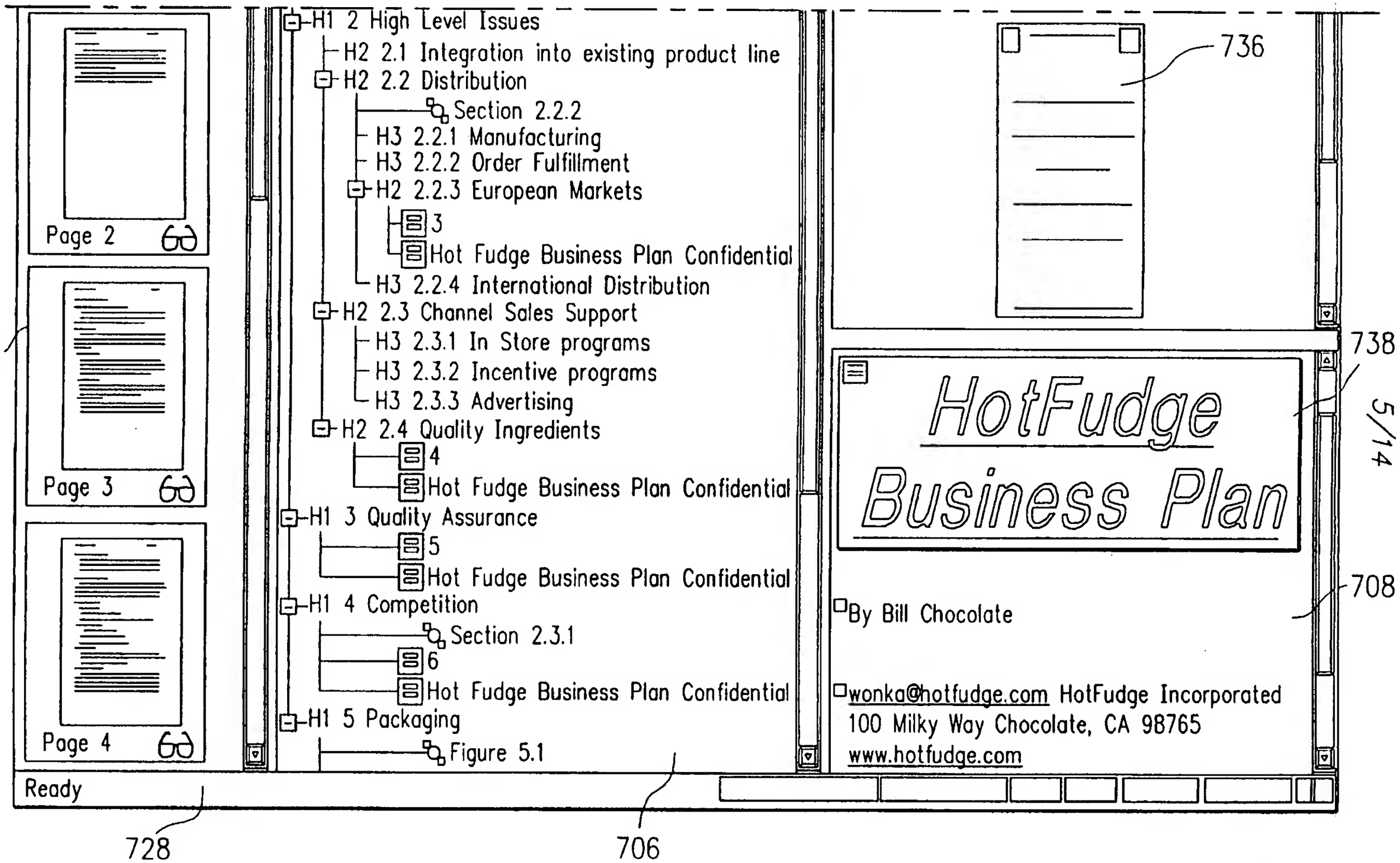

Page 2
Page 3
Page 4
702
H1 2 High Level Issues
H2 2.1 Integration into existing product line
H2 2.2 Distribution
Section 2.2.2
H3 2.2.1 Manufacturing
H3 2.2.2 Order Fulfillment
H2 2.2.3 European Markets
3
Hot Fudge Business Plan Confidential
H3 2.2.4 International Distribution
H2 2.3 Channel Sales Support
H3 2.3.1 In Store programs
H3 2.3.2 Incentive programs
H3 2.3.3 Advertising
H2 2.4 Quality Ingredients
4
Hot Fudge Business Plan Confidential
H1 3 Quality Assurance
5
Hot Fudge Business Plan Confidential
H1 4 Competition
Section 2.3.1
6
Hot Fudge Business Plan Confidential
H1 5 Packaging
Figure 5.1
736
738
HotFudge
Business Plan
708
By Bill Chocolate
wonka@hotfudge.com HotFudge Incorporated
100 Milky Way Chocolate, CA 98765
www.hotfudge.com
Ready
728
706

FIG. 7B

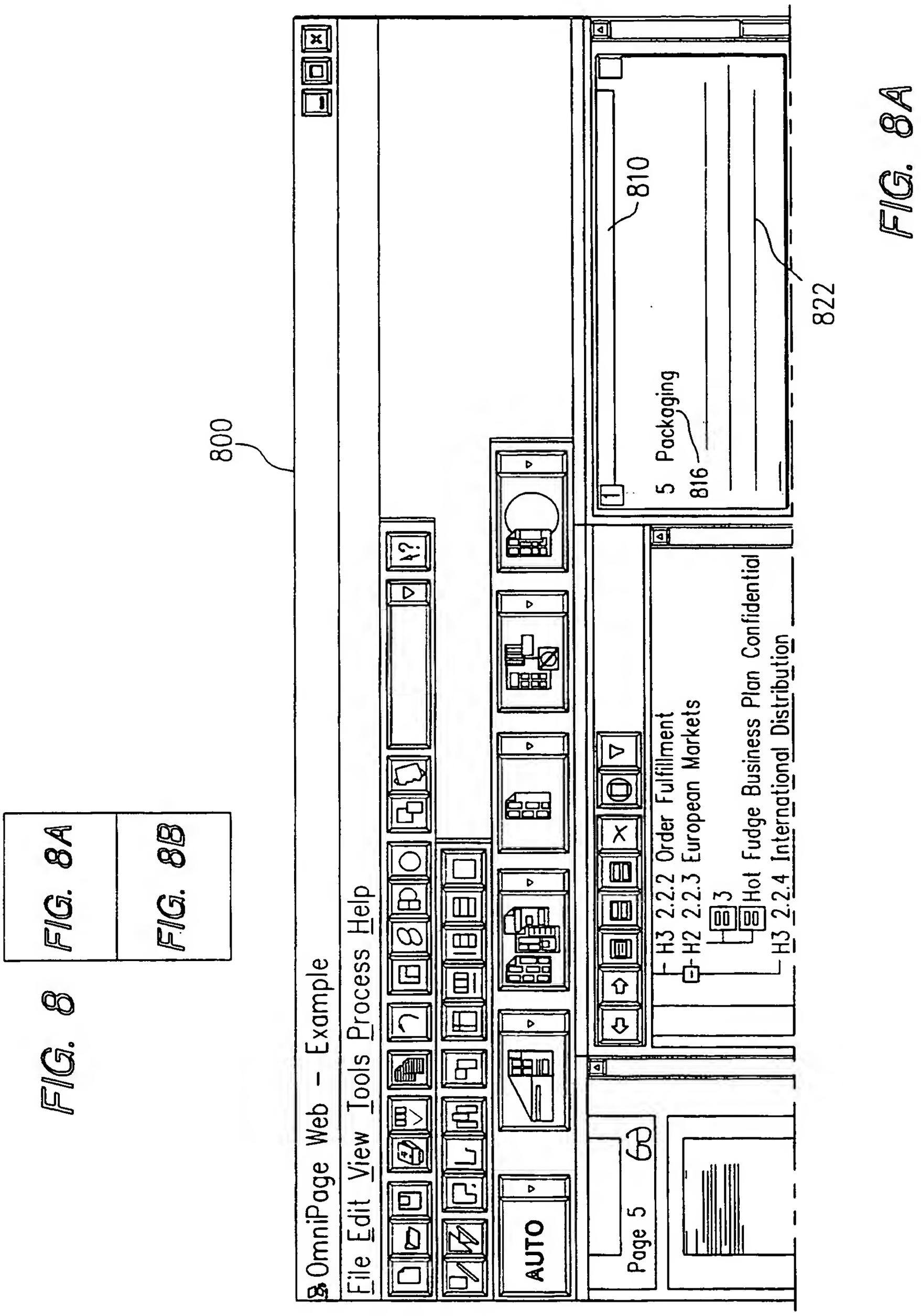


FIG. 8A

FIG. 8B

FIG. 8

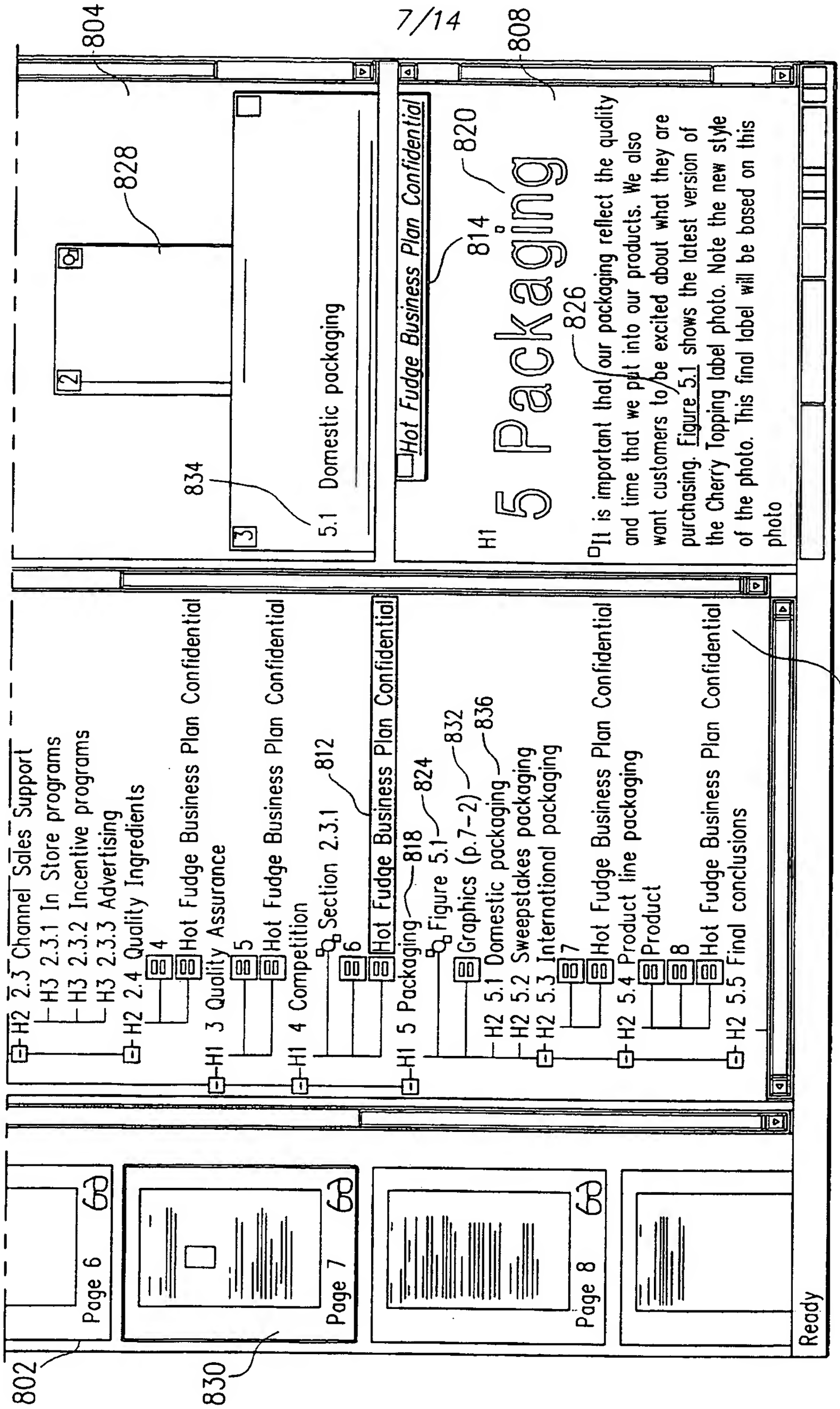

804
808
828
820
814
Packaging
826
Hot Fudge Business Plan Confidential
834
5.1 Domestic packaging
H1 5
It is important that our packaging reflect the quality and time that we put into our products. We also want customers to be excited about what they are purchasing. Figure 5.1 shows the latest version of the Cherry Topping label photo. Note the new style of the photo. This final label will be based on this photo
H2 2.3 Channel Sales Support
H3 2.3.1 In Store programs
H3 2.3.2 Incentive programs
H3 2.3.3 Advertising
H2 2.4 Quality Ingredients
4
Hot Fudge Business Plan Confidential
H1 3 Quality Assurance
5
Hot Fudge Business Plan Confidential
H1 4 Competition
Section 2.3.1
812
6
Hot Fudge Business Plan Confidential
H1 5 Packaging
818
824
Figure 5.1
832
Graphics (p.7-2)
836
H2 5.1 Domestic packaging
H2 5.2 Sweepstakes packaging
H2 5.3 International packaging
7
Hot Fudge Business Plan Confidential
H2 5.4 Product line packaging
Product
8
Hot Fudge Business Plan Confidential
H2 5.5 Final conclusions
806
Page 6
Page 7
Page 8
Ready
802
830

FIG. 8B

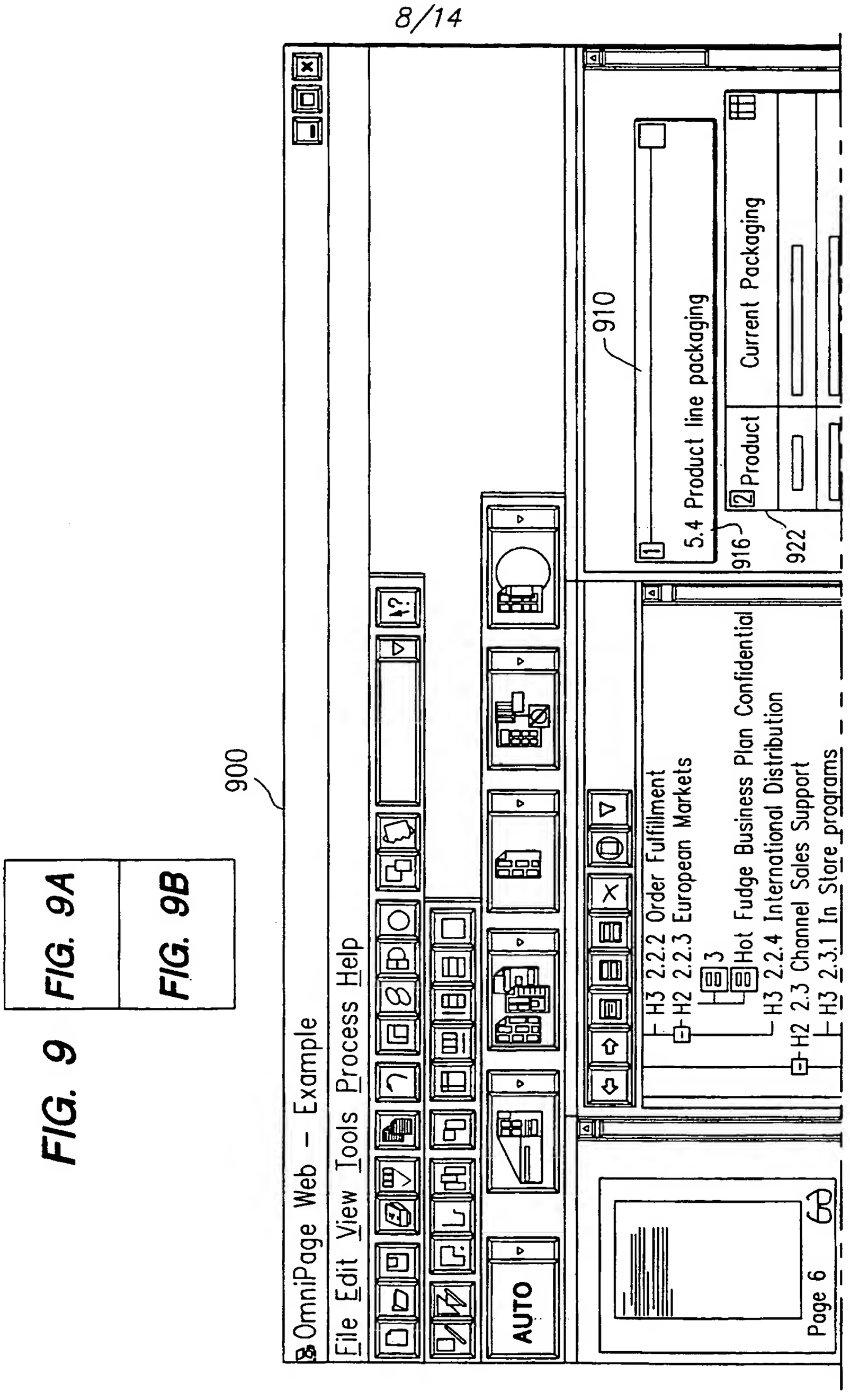


FIG. 9 FIG. 9A FIG. 9B

FIG. 9A

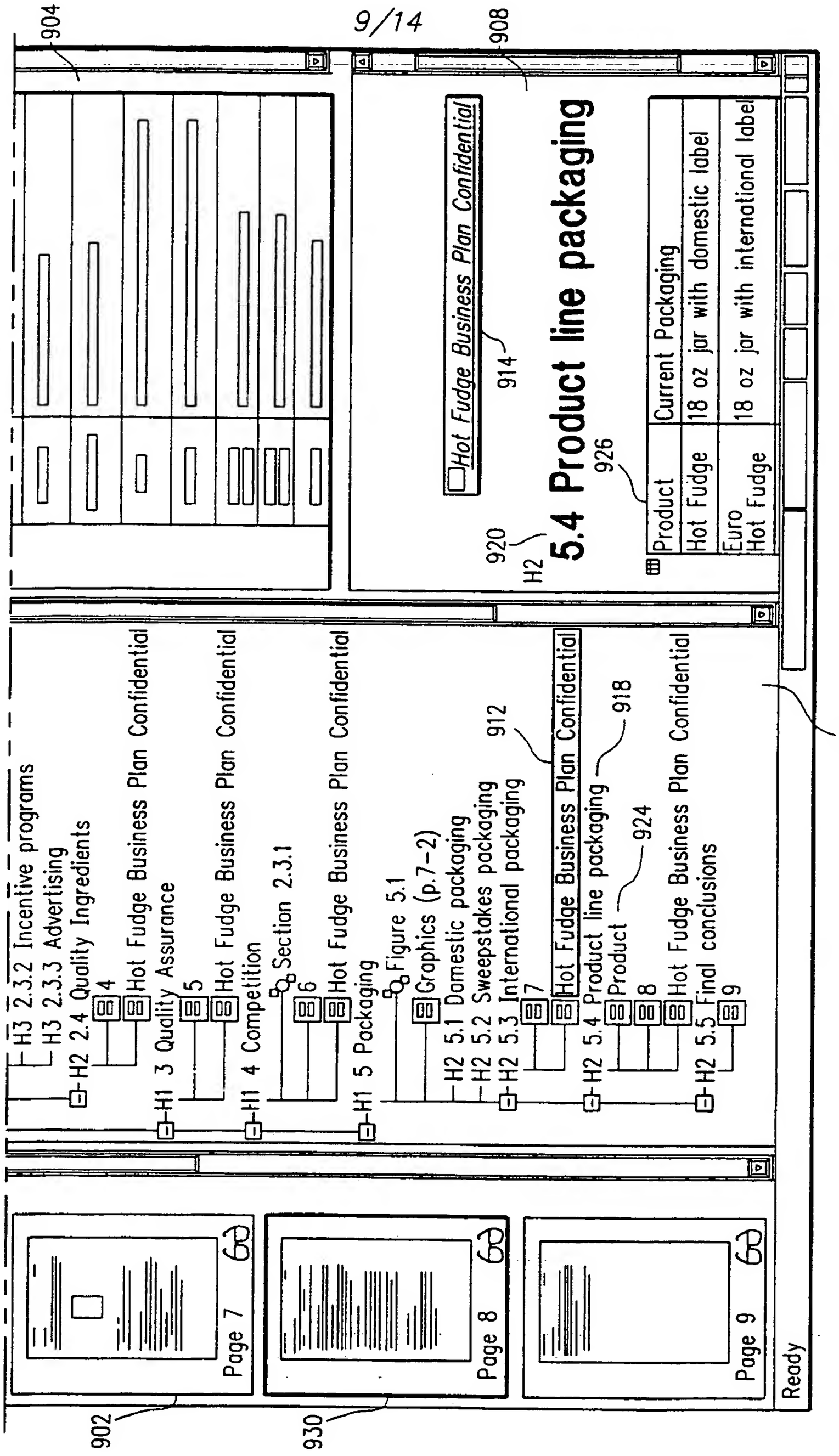

904
908
914
Hot Fudge Business Plan Confidential
5.4 Product line packaging
926
920
H2
Current Packaging
18 oz jar with domestic label
18 oz jar with international label
Product
Hot Fudge
Euro Hot Fudge
H3 2.3.2 Incentive programs
H3 2.3.3 Advertising
H2 2.4 Quality Ingredients
4
Hot Fudge Business Plan Confidential
H1 3 Quality Assurance
5
Hot Fudge Business Plan Confidential
H1 4 Competition
Section 2.3.1
6
Hot Fudge Business Plan Confidential
H1 5 Packaging
Figure 5.1
Graphics (p.7-2)
H2 5.1 Domestic packaging
H2 5.2 Sweepstakes packaging
912
H2 5.3 International packaging
7
Hot Fudge Business Plan Confidential
918
H2 5.4 Product line packaging
Product
924
8
Hot Fudge Business Plan Confidential
H2 5.5 Final conclusions
9
906
Page 7
Page 8
Page 9
Ready
902
930

FIG. 9B

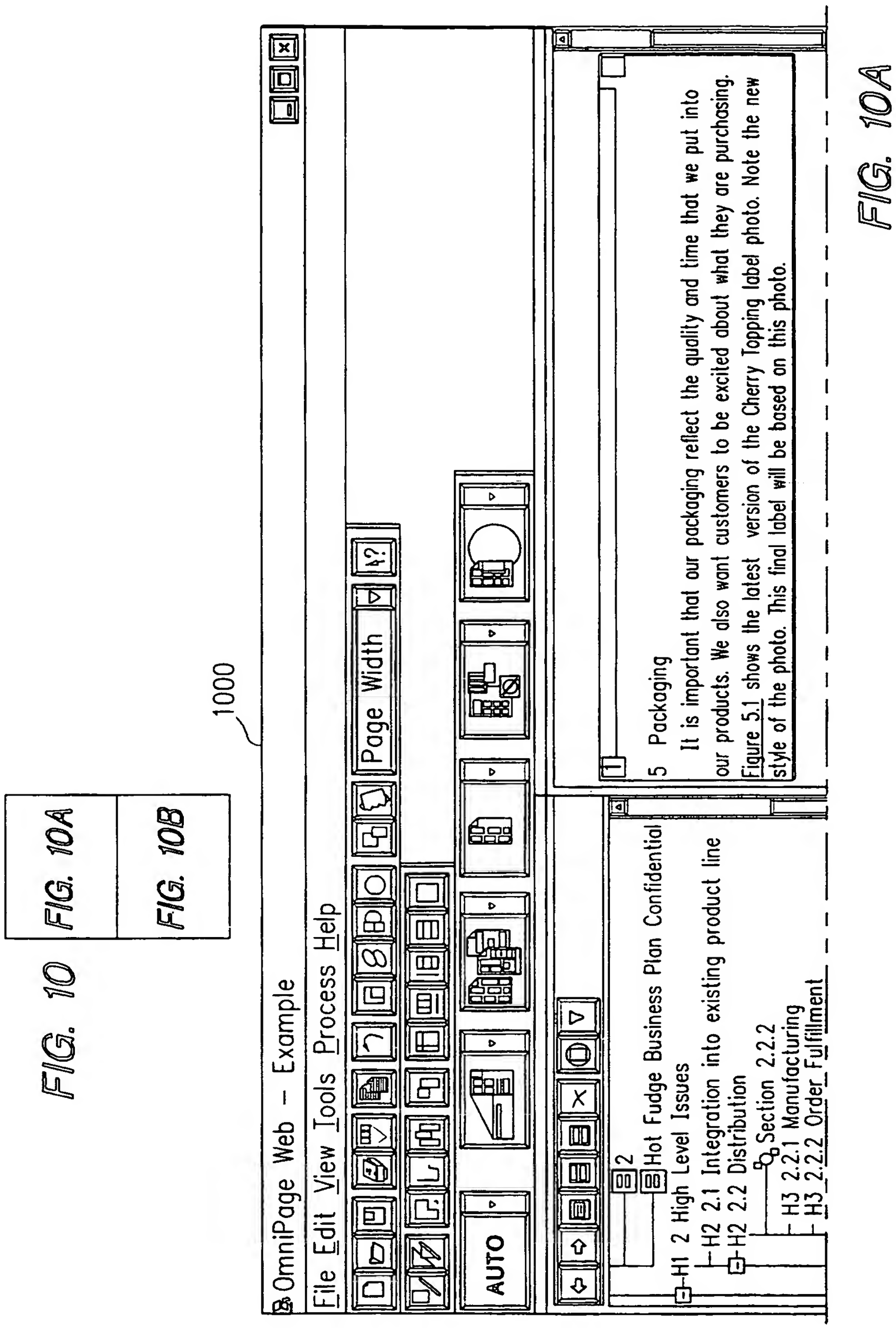


FIG. 10A

FIG. 10B

FIG. 10

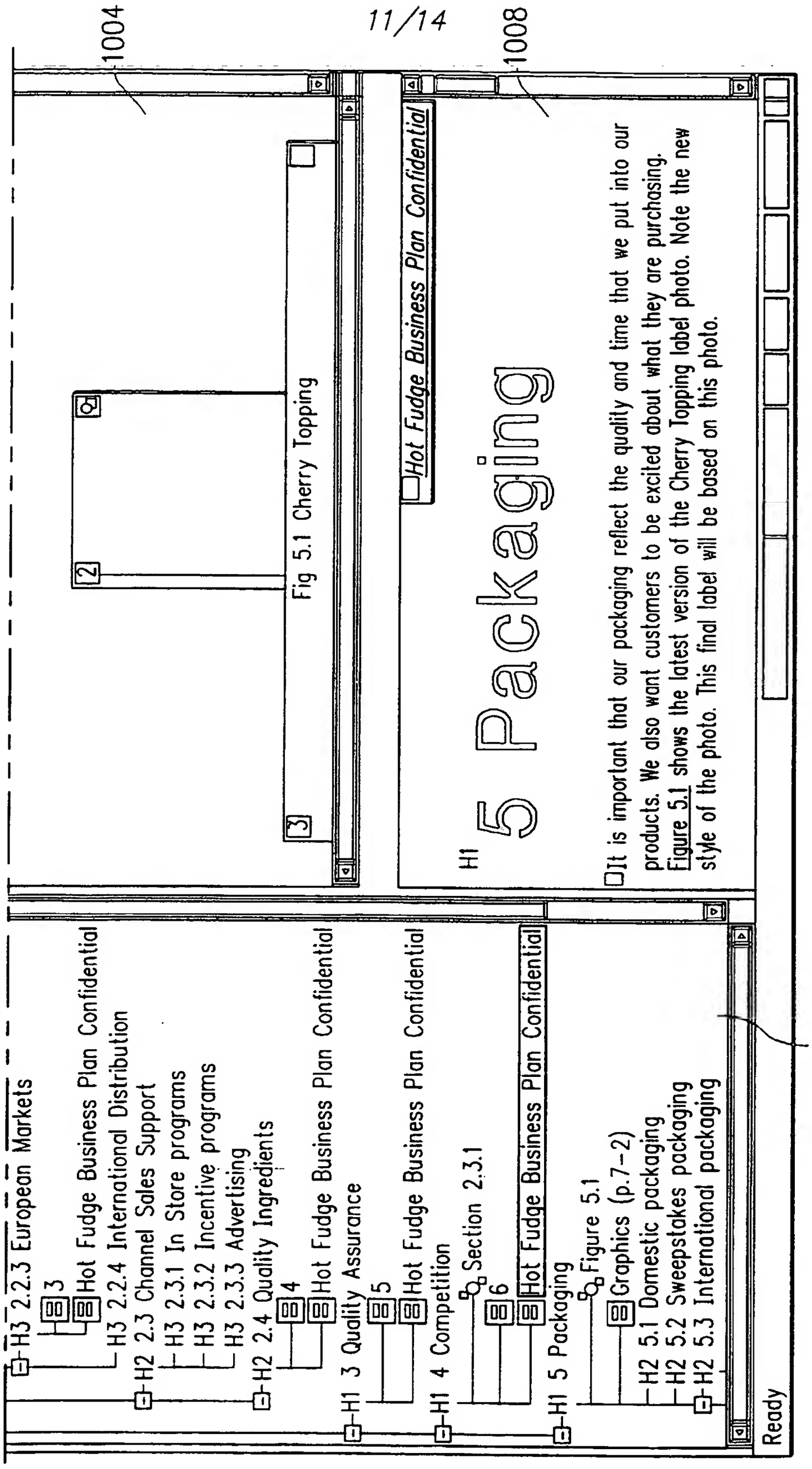

1004
1008
Hot Fudge Business Plan Confidential
Cherry Topping
[2]
[3]
Fig 5.1 Ch
H1 5 Packaging
It is important that our packaging reflect the quality and time that we put into our products. We also want customers to be excited about what they are purchasing. Figure 5.1 shows the latest version of the Cherry Topping label photo. Note the new style of the photo. This final label will be based on this photo.
H3 2.2.3 European Markets
3
Hot Fudge Business Plan Confidential
H3 2.2.4 International Distribution
H2 2.3 Channel Sales Support
H3 2.3.1 In Store programs
H3 2.3.2 Incentive programs
H3 2.3.3 Advertising
H2 2.4 Quality Ingredients
4
Hot Fudge Business Plan Confidential
H1 3 Quality Assurance
5
Hot Fudge Business Plan Confidential
H1 4 Competition
Section 2.3.1
6
Hot Fudge Business Plan Confidential
H1 5 Packaging
Figure 5.1
Graphics (p.7-2)
H2 5.1 Domestic packaging
H2 5.2 Sweepstakes packaging
H2 5.3 International packaging
1006
Ready

FIG. 10B

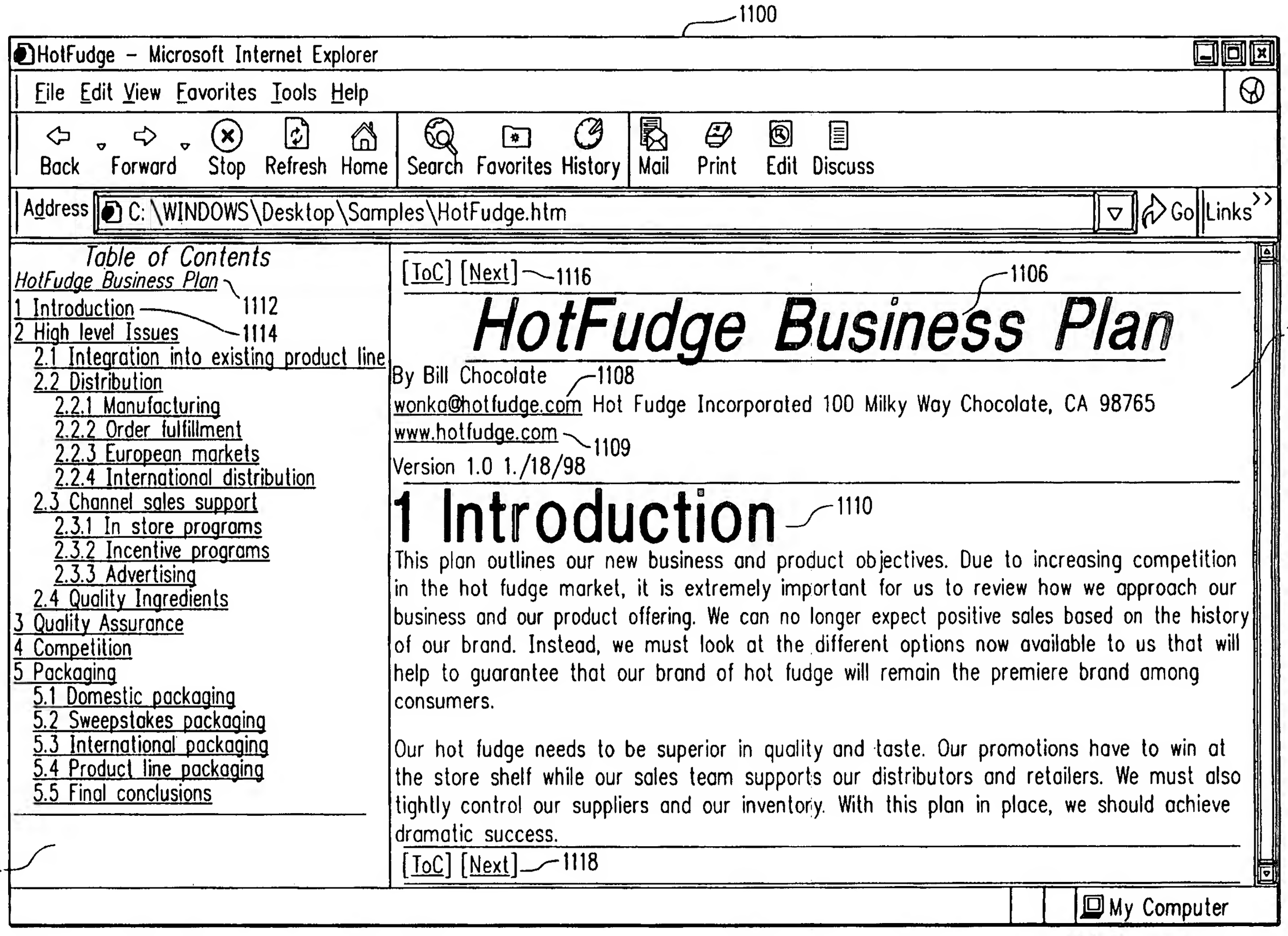

1100
HotFudge – Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Forward Stop Refresh Home Search Favorites History Mail Print Edit Discuss
Address C:\WINDOWS\Desktop\Samples\HotFudge.htm
Go Links
Table of Contents
HotFudge Business Plan
1 Introduction 1112
2 High level Issues 1114
2.1 Integration into existing product line
2.2 Distribution
2.2.1 Manufacturing
2.2.2 Order fulfillment
2.2.3 European markets
2.2.4 International distribution
2.3 Channel sales support
2.3.1 In store programs
2.3.2 Incentive programs
2.3.3 Advertising
2.4 Quality Ingredients
3 Quality Assurance
4 Competition
5 Packaging
5.1 Domestic packaging
5.2 Sweepstakes packaging
5.3 International packaging
5.4 Product line packaging
5.5 Final conclusions
1104
[ToC] [Next] 1116
1106
HotFudge Business Plan
1102
By Bill Chocolate 1108
wonka@hotfudge.com Hot Fudge Incorporated 100 Milky Way Chocolate, CA 98765
www.hotfudge.com 1109
Version 1.0 1./18/98
1 Introduction 1110
This plan outlines our new business and product objectives. Due to increasing competition in the hot fudge market, it is extremely important for us to review how we approach our business and our product offering. We can no longer expect positive sales based on the history of our brand. Instead, we must look at the different options now available to us that will help to guarantee that our brand of hot fudge will remain the premiere brand among consumers.
Our hot fudge needs to be superior in quality and taste. Our promotions have to win at the store shelf while our sales team supports our distributors and retailers. We must also tightly control our suppliers and our inventory. With this plan in place, we should achieve dramatic success.
[ToC] [Next] 1118
My Computer

FIG. 11

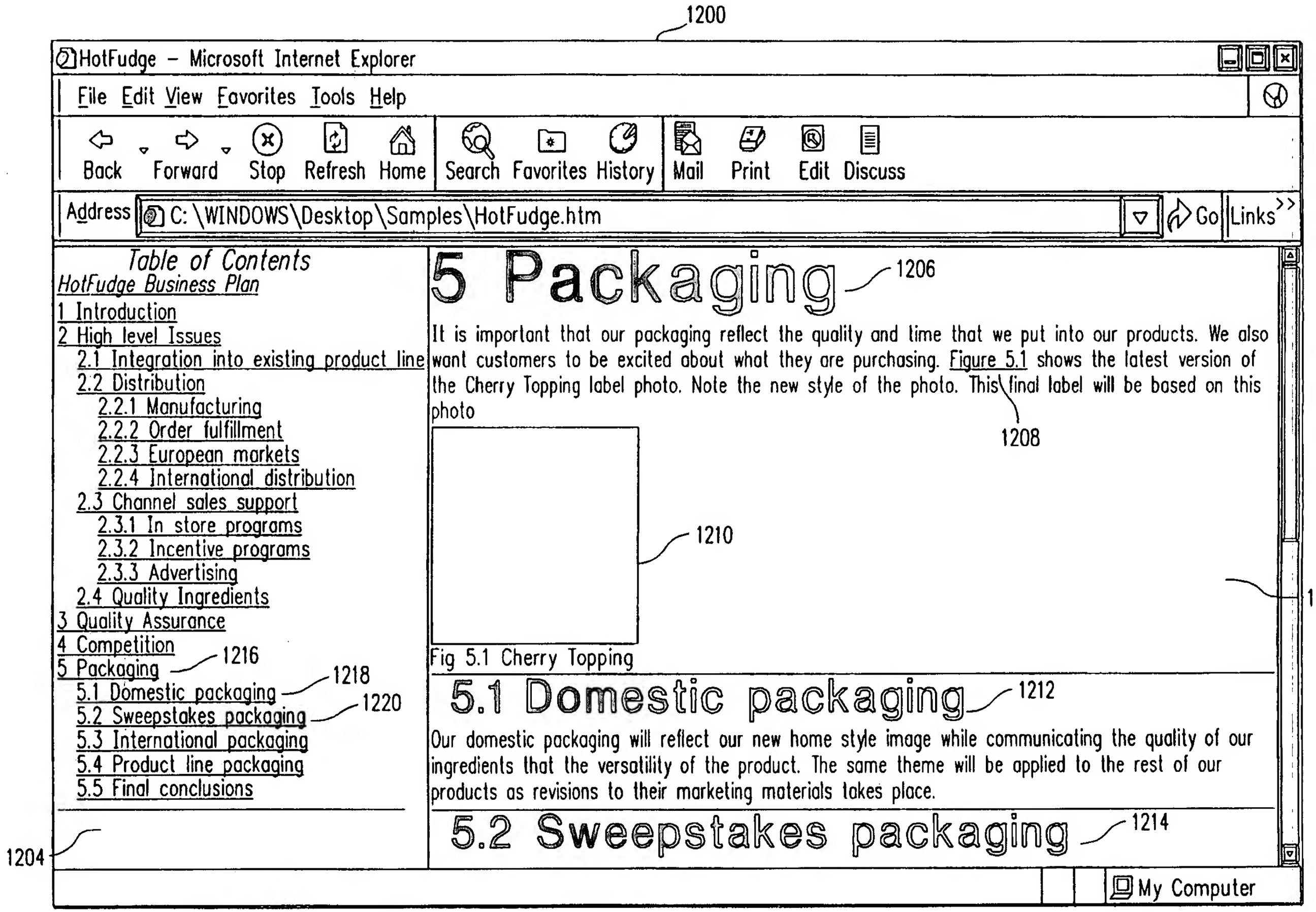


FIG. 12

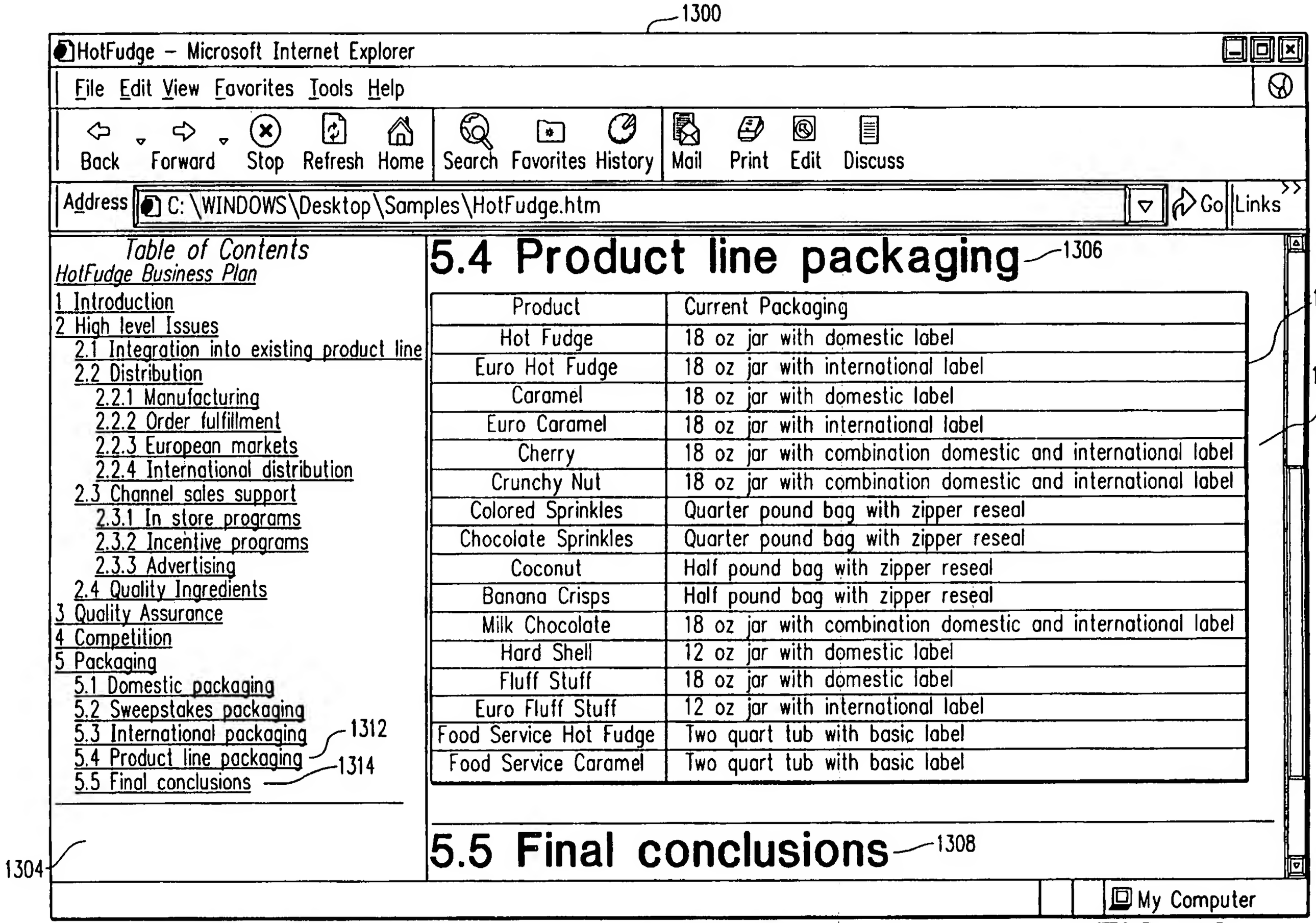

1300
HotFudge – Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Forward Stop Refresh Home Search Favorites History Mail Print Edit Discuss
Address C:\WINDOWS\Desktop\Samples\HotFudge.htm
Go Links
Table of Contents
HotFudge Business Plan
1 Introduction
2 High level Issues
2.1 Integration into existing product line
2.2 Distribution
2.2.1 Manufacturing
2.2.2 Order fulfillment
2.2.3 European markets
2.2.4 International distribution
2.3 Channel sales support
2.3.1 In store programs
2.3.2 Incentive programs
2.3.3 Advertising
2.4 Quality Ingredients
3 Quality Assurance
4 Competition
5 Packaging
5.1 Domestic packaging
5.2 Sweepstakes packaging
5.3 International packaging
5.4 Product line packaging
5.5 Final conclusions
1312
1314
1304
5.4 Product line packaging
1306
1310
1302
Product | Current Packaging
Hot Fudge | 18 oz jar with domestic label
Euro Hot Fudge | 18 oz jar with international label
Caramel | 18 oz jar with domestic label
Euro Caramel | 18 oz jar with international label
Cherry | 18 oz jar with combination domestic and international label
Crunchy Nut | 18 oz jar with combination domestic and international label
Colored Sprinkles | Quarter pound bag with zipper reseal
Chocolate Sprinkles | Quarter pound bag with zipper reseal
Coconut | Half pound bag with zipper reseal
Banana Crisps | Half pound bag with zipper reseal
Milk Chocolate | 18 oz jar with combination domestic and international label
Hard Shell | 12 oz jar with domestic label
Fluff Stuff | 18 oz jar with domestic label
Euro Fluff Stuff | 12 oz jar with international label
Food Service Hot Fudge | Two quart tub with basic label
Food Service Caramel | Two quart tub with basic label
5.5 Final conclusions
1308
My Computer

FIG. 13

INTERNATIONAL SEARCH REPORT

rnational Application No
PCT/US 01/03557

A. CLASSIFICATION OF SUBJECT MATTER
IPC 7 G06K9/00 G06F17/30

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
IPC 7 G06K

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practical, search terms used)
EPO-Internal

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category °	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	EP 0 843 276 A (CANON INFORMATION SYST INC) 20 May 1998 (1998-05-20) page 5, line 9 - line 41 page 11, line 35 -page 13, line 41; figure 14	1-9, 18-25, 31-38
X A	US 5 963 966 A (WHALEN MARGARET J ET AL) 5 October 1999 (1999-10-05) abstract column 9, line 8 - line 17	1,18,31, 36 43
X	GANN R: "CAERE IMPROVES ITS OCR INTERFACE" PC USER, 21 August 1996 (1996-08-21), XP002055985 the whole document	1,18,31, 36

-/--

[X] Further documents are listed in the continuation of box C.

[X] Patent family members are listed in annex.

° Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier document but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art.

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
11 July 2001	17/07/2001

Name and mailing address of the ISA	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL – 2280 HV Rijswijk Tel. (+31–70) 340–2040, Tx. 31 651 epo nl, Fax: (+31–70) 340–3016	Sonius, M

1

Form PCT/ISA/210 (second sheet) (July 1992)

INTERNATIONAL SEARCH REPORT

International Application No
PCT/US 01/03557

C.(Continuation) DOCUMENTS CONSIDERED TO BE RELEVANT

Category °	Citation of document, with indication,where appropriate, of the relevant passages	Relevant to claim No.
X	GANN R: "ACCURATE OCR FOR COMPLEX PAGES" PC USER, 2 October 1996 (1996-10-02), XP002055986 the whole document ---	1,18,31, 36
A	US 5 781 914 A (STORK DAVID G ET AL) 14 July 1998 (1998-07-14) abstract ---	1,18,31, 36
A	YUAN Y ET AL: "Automatic document processing: A survey" PATTERN RECOGNITION,US,PERGAMON PRESS INC. ELMSFORD, N.Y, vol. 29, no. 12, 1 December 1996 (1996-12-01), pages 1931-1952, XP004015742 ISSN: 0031-3203 the whole document -----	1-43

Form PCT/ISA/210 (continuation of second sheet) (July 1992)

INTERNATIONAL SEARCH REPORT

Information on patent family members

ernational Application No
rCT/US 01/03557

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
EP 0843276 A	20-05-1998	US 5893127 A	06-04-1999
		JP 10162003 A	19-06-1998
US 5963966 A	05-10-1999	NONE	
US 5781914 A	14-07-1998	JP 9044383 A	14-02-1997